APPLICATION OF DISPLAY IN FLIGHT VEHICLE MISSION PERFORMANCE EVALUATION

Willi Konrad Alois Krauss

United States
Naval Postgraduate School

THESIS

APPLICATION OF DISPLAY
IN
FLIGHT VEHICLE MISSION PERFORMANCE EVALUATION

by

Willi Konrad Alois Krauss

October 1969

Application of Display in Flight Vehicle Mission Performance Evaluation

by

Willi Konrad Alois Krauss
Kapitänleutnant, Federal German Navy

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN ELECTRICAL ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL
October 1969

ABSTRACT

With the increase in speed of moving vehicles there is a growing need for rapid, easy to interprete display of information concerning its dynamic state.

With growing equipment stability a remedy can grow out of research and application of computer aided information presentation, especially the production of proper displays. While all information can be made to correspond to familiar schemes as close as desired new approaches to the general problem must be sought to find an improved information presentation. This study is concerned with the application of computer generated graphics for one or more display consoles for aid in the operation of the vehicle. In the first chapter a thought experiment is performed and evaluated and a program structure for the operation of a flight vehicle is described. In the second chapter a major aspect of the general program is chosen for a demonstration: a 'flight vehicle to destination' situation is subjected to an experimental investigation.

A general purpose computer with a graphics terminal that has become available just recently has been chosen for the study. Thereby implicitly a measure of the capability to perform a real time on line job is obtained.

TABLE OF CONTENTS

I. INTRODUCTION

Flight vehicles including a human operator are made to respond to his controls. A standard way to present the information necessary for the operation of the particular flight vehicle has developed. Different vehicles (commerical aircraft, military aircraft, or even a submarine) have their favourable way to show the important data concerning the dynamic state of the vehicle. For the individual type there exist certain situations, in which control forces applied will be based on information which might probably be of relevance in this one situation only.

The traditional way to present the necessary information is to enumerate all situations and provide a set of meters and indicators for each. Thus for n situations there will be n sets of meters. The design policy tries to use one particular meter in as many situations as possible and suitable. This procedure has several disadvantages; among others:

(1) Supervision is made difficult by "clutter" effects the operator cannot or only to a small degree overcome by selective techniques, if n is large.

(2) The information presented will often require further imaginative operator's work, i.e. the presentation is often far from optimal due to indirectness.

(3) The qualification of an operator must meet high requirements: combinatorial speed as well as spacial imaginative power. The last two are of special importance in aircraft operation. There will always be long training periods.

It has been claimed that a digital computer having a digital CRT as a peripheral device can take over almost all the combinatorial work in a complex situation, leaving the true supervisory task to the operator:

(a) No 'clutter' effects need remain because the operator has a large range of display options; all non-relevant information is suppressed.

(b) The information presented can be made optimal by suitably transforming and evaluating measured data, avoiding any unwanted indirectness.

(c) The requirements on spacial imaginative talent of the operator can be lowered by providing three-dimensional pictures of a simplified ('filtered') geometry, having no anticues.

An application of display techniques in flight vehicle mission performance evaluation requires that the problem of presenting state information is restated and new ways to solve it are sought. Given the tremendous flexibility of a high speed digital computer it is reasonable to aim at more than a true copy of existing equipment. Some information will always be presented in one dimensional form (e.g. a gyro compass). The capability of a digital CRT to allow information to be presented in meter form will be demonstrated in Chapter III. So far only limited use of the human capability to orient in three dimensions has been possible. A reason for this is the lack of convincing equipment. Contact analog devices ('Conalog'), see ref. 9, synthesize a three dimensional structure from measured data. Use of 'Conalog' indicators has been made with success. They are a remedy only in the absence of a true perspective picture of the environment. If true perspective pictures in any direction

can be generated the 'Conalog' presentation will be possible as a by-product, although still very useful.

Three dimensional displays combined with standard presentations are regarded but a first step in the direction of new information presentation. (An example has been demonstrated in Chapter III.) A basic requirement for a digitally controlled information presentation is the existence of a managing program. In Chapter II, fundamental aspects of such a program are investigated.

II. PRELIMINARY CONSIDERATIONS

A. MANAGEMENT PROGRAM STRUCTURE

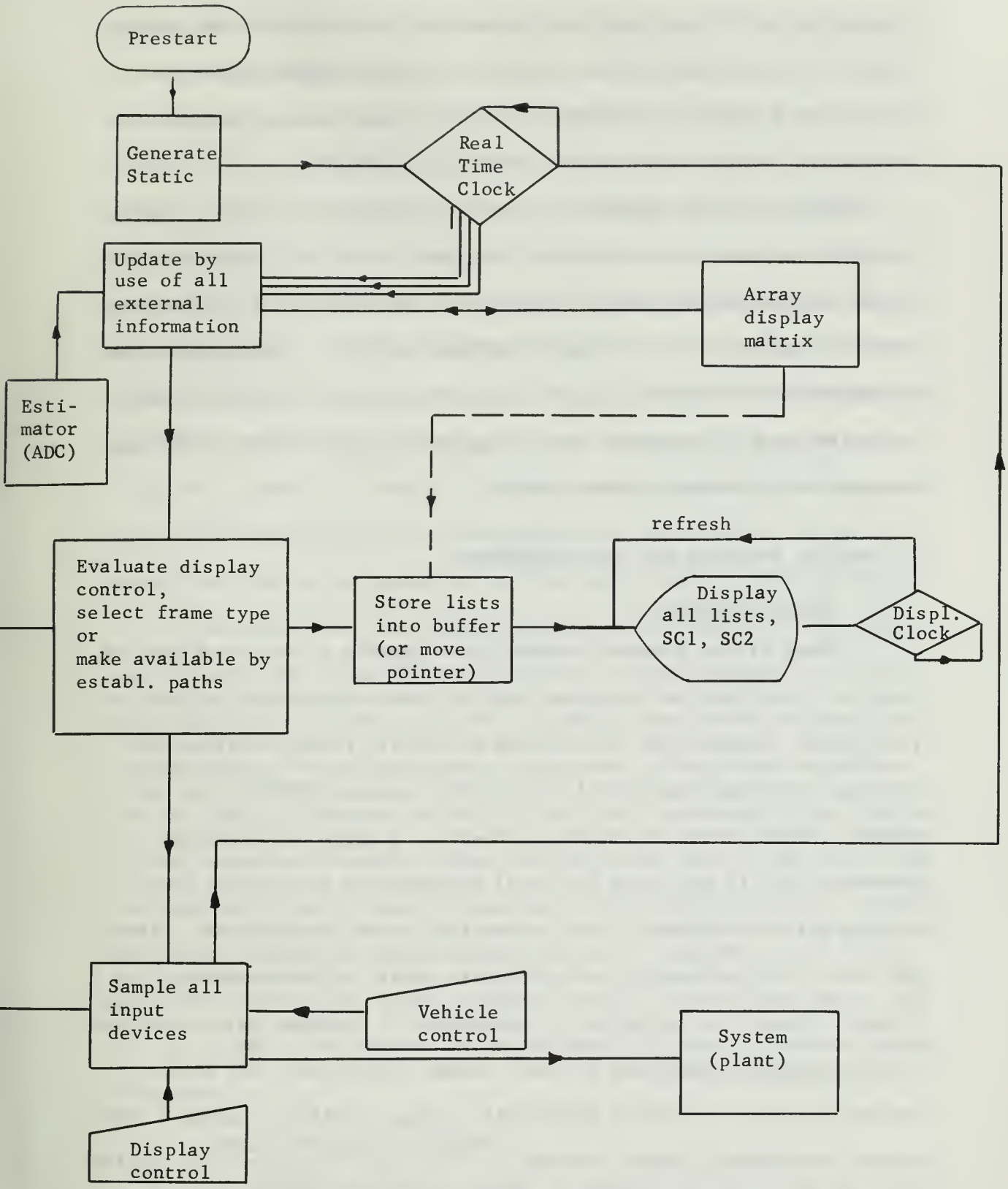
In the following the structure of a program (that could run on an arbitrary present day general purpose computer) is described in block diagram form. See fig. 1.

B. DESCRIPTION OF BLOCK DIAGRAM

Updating of all the variables that describe the state of the system and its references must be done at precise intervals. It also must be completed within time limits.

A real time clock will initiate analog to digital conversion (ADC) at the precise times, following which the transforming mathematical operations will be carried out. Several levels of interrupts and program traps are necessary. Levels are indicated by multiple arrows leading to the block called UPDATE. The main clock interval is determined by the control requirements as well as the plant response. Increased accuracy requires prediction which in turn requires higher sampling rates. Those problems are of no concern here as long as the computational speed of the computer is sufficient.

For the storage and updating of the display matrix see under C. Under 'EVALUTE DISPLAY CONTROL' the proper programming paths are set up that establish the connection between the display matrix (which contains all available data-variables as well as constants-in transformed form) and the storage routine, which transfers as the next step the selected data into the display buffer. Depending on the type of display chosen by the operator a projecting routine converting three



BLOCK DIAGRAM OF MANAGING PROGRAM + ROUTINES

FIGURE 1

dimensional coordinates into two dimensional coordinates of the focal plane will be inserted in the path from storage (display matrix) to the display buffer. The display clock determines the time when to refresh the vectors drawn with the data in the buffer.

'SAMPLE ALL INPUT DEVICES' performs two independent tasks: one type of inputs is meant as controls for the plant, while the other determines if and what change of display is desired. The latter part thereby provides the parameters for 'EVALUATE DISPLAY CONTROL'. After completion of sampling the computer returns to an idling state or whatever batch processing type of a job has been assigned (e.g. postflight evaluation preparation by storing certain data).

C. DISPLAY POLICIES AND DATA STRUCTURES

1. List Processing

Most of the graphics terminals are capable of accepting data in forms of lists that are displayed item by item sequentially by means of interrupts. Certain bits in the data or certain register values controlling vector generation will determine a blanked (MOVE) or an unblanked (DRAW) motion of the beam. Whenever a separate refreshing mechanism that is not using the vital registers of the central processing unit is available, list processing is the preferred way. (Note: The term 'list processing' as used herein should be distinguished from a common usage, i.e. referring to techniques of scanning tree organized symbol structures and using chained storage allocation. The lists referred to here are simply sequential strings of micro-commands.) The number of refreshing passes through the list must be appreciably greater than the number of changes made in the list. Hence it is preferable in terms of computing time minimization (or brightness improvement) to

generate a list which in turn is displayed instead of building up a picture as it is made up of vectors, processing them sequentially as soon as they are computed. If no list can be established, it is likely that an improper information presentation has been chosen. (E.g. if a rotating vector is displayed and the angular velocity is so high that the whole circle area is illuminated, there is no longer any position information presented which can be visually monitored!) Proper display must conform with the plant dynamic control. In cases where noise is superimposed on a vector, a smear out will result which will be resolved by the eye into an optical average. It is recommended, however, to insert an optimal estimator if there is enough computational time available and analog averages are unsatisfactory. It violates the policy of presenting clutterfree graphics to allow noisy vectors to be displayed.

From the above it follows that data for display are constants over an extended period of time (in terms of machine time). All available data should already be stored in usable form before the real time process starts. Read only memory can be used if the data is constant for all time. A fraction of the data will vary permanently and requires central processor storage. These data are those used in the projecting routines which are in general functions of the flight vehicle's position in space. Permanently constant data are those comprising the static part of an indicator: scale, numbers, radial lines, titles, etc. Only one or at most a few numbers per meter need high speed read/write memory locations.

2. Compute and Draw Procedure

A different situation exists if there is no way to refresh while update and sampling computation is carried out. In this case the display

clock will start the update operation at the end of a selected number of refreshing paths. The total managing work will be distributed over several clock pulses. Unless the number of vectors is small (as is the case in the programming project of Chapter III), list processing is still advantageous. A trade off must be made between the average computation time per vector and the time the program must be trapped while the vectors are drawn (times required to store and modify will add to the overall list processing time). This latter part will be further discussed in Chapter III.

D. PROPOSALS FOR DISPLAY IN A SPECIAL CASE

Since the flight vehicle dynamics determine its most suitable information presentation for the purpose of operation, it is difficult to generalize when configurations for one or more display scopes are discussed. If the flight vehicle is identified as an aircraft and some departure from standard presentations is allowed, a typical situation can be imagined. See ref. 10. If the proposal for a display configuration leaves room for changes and amendments, it is justified for the purpose of clarity to look at a particular situation. Since flexibility is one of the attractions a digitally controlled CRT has, certain conclusions which will hold for many cases can be drawn. By describing what can be done in a particular case and by investigating the requirements, a scheme is set up that can be used in a variety of cases.

1. Scope Presentation

Control of a flight vehicle can be based upon

(1) no further information than a set of meters,

(2) visual observation of the surroundings and a few indicators.

The first neglects the built-in capacity of the human being to orient in three dimensions if a proper picture is presented. The second has the disadvantage that the visual conditions will not always exist and/or the time available is not sufficient to make out the given cues.

It is proposed to construct a true perspective picture of a known environment with the help of a digital computer. The perspective view shall be the main information for the operator to base his control decision on. The knowledge of the enviroment could come from a topographical map and could be aided by radar measurements and telemetry to ascertain relevance. If there is lack of tabular (mapped) information, the picture on the scope could be constructed from electronic measurements only. Combinations of 'Conalog' elements and real perspective pictures should be left at the operator's choice, also whatever indicator is desired to observe closely. Since the number of scopes will always be limited some control over the projecting subroutine should be given to the operator: e.g. change from normal to wide angle projection (wide angle photography) to include a larger sector of the surroundings.

2. Picture Generation

The task of the computer would be to

(1) generate a correlated set of three dimensional coordinates, i.e. evaluate a set of vectors in three dimensions,

(2) project these three dimensions on a focal plane,

(3) display the vectors obtained in focal plane coordinates.

The picture generated by this method may not contain massive anticues and should suppress hidden lines if necessary. Since this picture is permanently updated when the flight vehicle moves, it provides an excellent method to aid the space orientation of an operator. See ref. 15.

Programs to project a general set of vectors in three dimensions suppressing hidden lines exist (ref. 11). In Chapter III the implementation of the general features (projection and updating) on the Adage graphics terminal is described, also a typical indicator is displayed simultaneously with the three dimensional object, projected in perspective.

The second type of display under the control of the operator is the display of meters with one or a few vectors indicating the variable values. It seems to be unnecessary to simulate completely a standard indicator as long as the same information is presented.

The configurations mentioned above can be combined easily. An interesting combination will result from Pseudo 3D included in real perspective projections. Proposals concerning this combination are made in Chapter III.

E. COMPUTATIONAL REQUIREMENTS, DATA REQUIREMENTS, LANGUAGE CONSIDERATIONS

Data requirements for a three dimensional transformation and projection need a study of its own and will not be discussed further. The storage requirements and operations on data comprising the two dimensional parameters for indicators are discussed in the following:

See figure 2. Step (1) is a read operation which is done prior to the real time operation of the system. Read in will be a matrix which contains in fixed point form the parameters necessary to generate a standard type of indicator:

- initial and final value of the variable
- number of radial lines for round or horizontal and vertical lines for rectangular meters

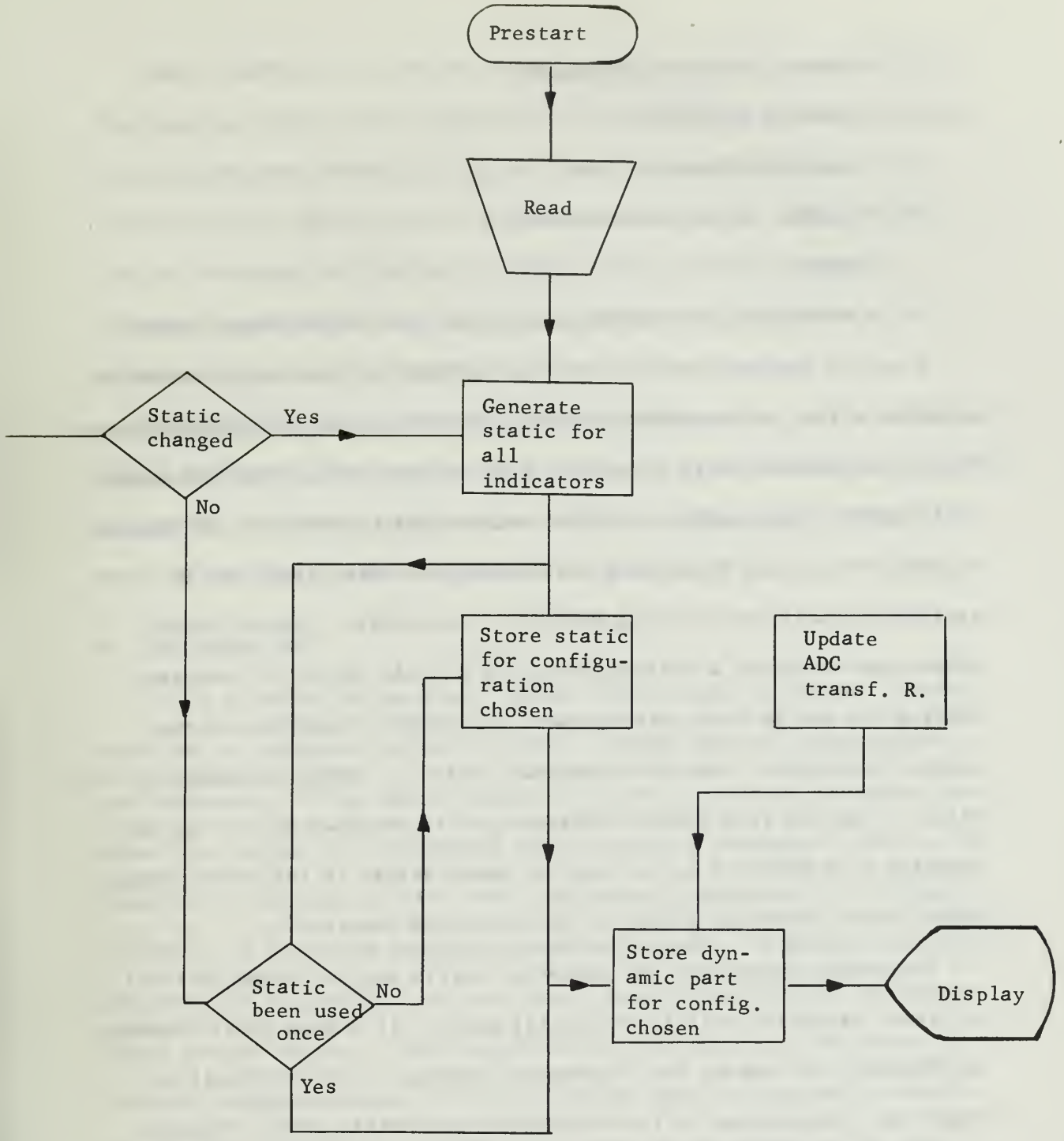

Prestart
Read
Static changed
Yes
No
Generate static for all indicators
Store static for configu-ration chosen
Update ADC transf. R.
Static been used once
No
Yes
Store dyn-amic part for config. chosen
Display

FIGURE 2

- shape, round or rectangular
- numbers to be shown
- spacing of radial lines
- linear or nonlinear spacing
- title
- configurations desired and offset for various cases, scale.

A set of routines must be provided to work on these input values to generate a list of starting and end points of vectors for drawing operations. In general every meter will have one set of vectors from which it is built. Only different offset values will be added to the vectors to make them usable in various configurations. The offset can be applied digitally or by using hardware if available. Before offset values can be added, a multiplication by a scale factor is necessary. Scaling too can be done with hardware or software depending on the machine available. From the coordinate value x, stored permanently, the value x' for use in a chosen configuration is obtained by solving the equation x' = deltax + scale times x, where deltax is the offset value. Scale is the reduction factor of the normalized quantity x.

The proper values for the parameter matrix must be found by trial and error after an intelligent initial guess. If a high level language, say FORTRAN, is used as the programming language it is difficult to regain all information if it is stored in one matrix only. Probably three or four matrices are necessary and a waste of storage will result. If, however, the prestart program can be written in assembly language, all information is most suitably packed into one matrix. The words can easily be evaluated in the prestart phase, most information being in YES or NO form and hence requiring one bit only.

Under PRESTART all data are computed and stored in a static array. The managing program will reference to the columns of the static array. It will authorize storage of chosen static (constant) data into the display buffer, where also the dynamic part is stored. The variable part of the meter is obtained in UPDATE, where analog to digital operation is triggered and the value referenced to the static array column of the particular quantity.

The composition of a display (called configuration) can be influenced from a keyboard. The operator will decide, if a prearranged configuration of several indicators together with perhaps a perspective graph or one peculiar one for his momentary needs is most suitable.

F. MODIFICATIONS

It is a matter of empirical design, which shape and visual impression for an indicator is most suitable. Human factor considerations are necessary. If no special copies of today's standard equipment are asked for (a special programming effort would be required), then it is possible to add and subtract easily different indicators for different variables by modifying the input parameter matrix. A general generation program will be possible for the static array if no special copies are asked for by the user. This might not be suitable for simulation of aircraft instrumentation. In this case the data of the static array have to be obtained in a different process, which could be the same digital CRT on which the meter is sketched with light pen and joystick. A general generation program, however, might be highly desired in an equipment testing situation where quick changes are often wanted.

G. SUMMARY

The center of vehicle mission performance evaluation with the aid of a digital computer and CRT is a managing program, around which subroutines are oriented:

- PRESTART, under which all static data are generated,
- UPDATE, where all variables are updated,
- SAMPLE INPUTS and EVALUATE; display control,
- transferring chosen data to display buffer and initiation of display.

The problem of acquisition of information about the geographical environment has not been tackled. In Chapter III such a structure is assumed to be given.

III. PROGRAMMING PROJECT

A. COMPUTATION AND DRAWING OF A VECTOR

This chapter describes a programming project which has been carried out to clarify and to provide with an experimental basis the considerations of Chapter II. The computer being used (ADAGE GRAPHICS TERMINAL) has a word length of 30 bits. The basic calculation is carried out in a 15 bit format. Almost all special instructions are laid out for half-word operation. The basic machine cycle is 2 microseconds. See ref. 5.

The capability of the AGT10 to have list display while other operations are carried out is provided. Since the number of vectors has been kept low, a procedure has been chosen where every vector is displayed immediately after finishing its computation. No interrupts are used at the start or at the end of a vector drawing. (See Appendix A on vector generation.) The program continues immediately after the vectors that control vector generation have been provided with the necessary values, control, scale, offset, intensity, xy-position. Refreshing and computation is done in one loop. This procedure allows change of data (due to rotation f.i.) per cycle, it is the most efficient way to generate a dynamic display. The minimum refreshing rate is 40 passes through the display list per second for no flicker; even if a picture is changing appreciably within a second still no 'jumps' will occur. If flickerless display can be established, it is possible to follow this procedure. For a dynamic display, however, it is not required to allow for changes at 40 times per second. A slower rate (less than 10 times

per second) will do. If the vector displayed represents the outcome of an electronic measurement, a noisy vector will be drawn if the measured quantity contains low frequency noise (which is typical because only low frequency quantities can be displayed).

No vector drawing, however, may begin before the old vector has been completed. This is no serious restriction. If computation is done between vector drawings, the vector generation can easily be controlled. To refresh at 40 times per second is a strong timing requirement for the overall program: updating and sampling, etc., must be distributed over several cycles to avoid flicker.

(1) All calculations must be carried out in minimum possible time. This excludes the use of floating point or double precision (30 bit) arithmetic.

(2) Storage must be used to trade for gain in time.

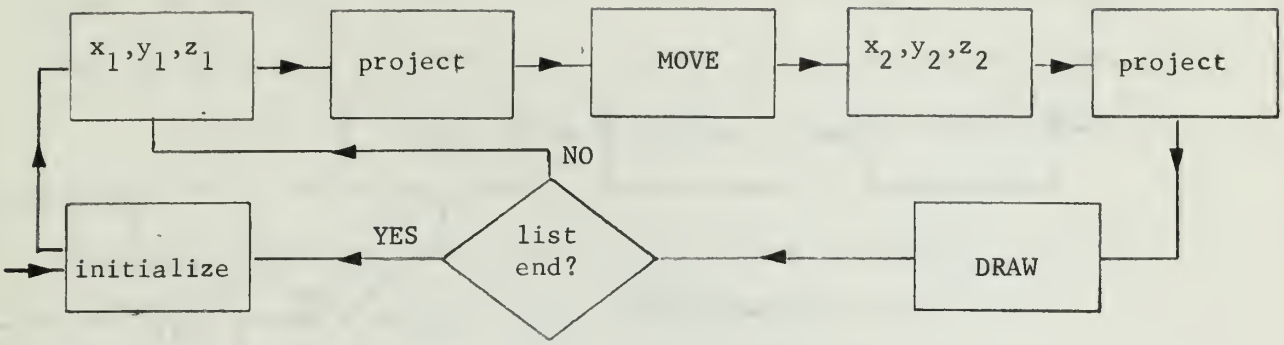
(3) No high level language can be allowed for the managing program or the projection/rotation routines. It is not possible to use a high level language efficiently in connection with word preparation for the various registers. Bit setting and checking is an often recurring operation. A remedy could be the insertion of a small assembly language routine.

B. GRAPH COMPLEXITY VERSUS VECTOR COMPUTATION-VECTOR DRAWING PROCEDURE

The graph complexity is measured by the number of vectors to be drawn. Drawing a computed vector allows two approaches:

1. Compute and Draw

   See flow diagram.

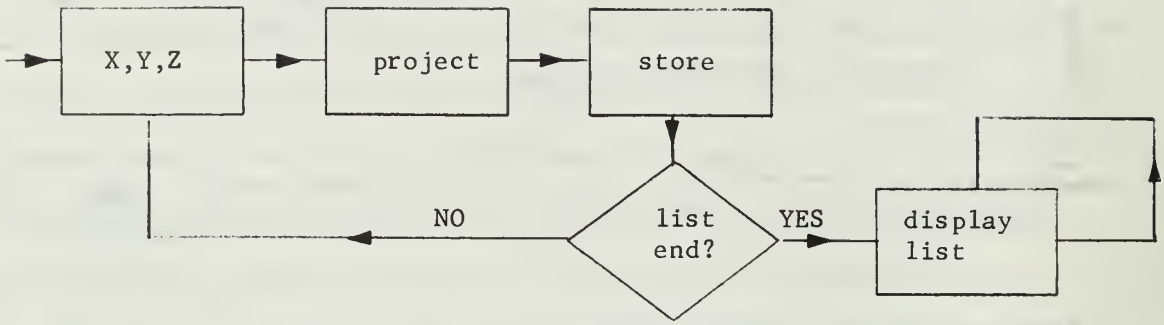


Take one xyz-set, project, MOVE; take next xyz-set, project, DRAW. Use mode bit control, no interrupts. (For further explanation see Appendix A.) Repeat this process as often as possible per second, at least 40 times.

As long as the computational work requires more than 30 microseconds, the vector drawing time is independent of the vector length. A projection of a vector (two xyz-sets) with the help of CVTXY (see Appendix C, program listings) has been determined to take 220 microseconds. If no other computational work is done, then flicker will occur as soon as a list requires more than 0.025 seconds. This is the case for 66 vectors. This is the number for the most favourable situation with no housekeeping of any kind required. If updating, etc., must be done the number of vectors will be reduced by the time required for housekeeping.

2. List Display

See flow diagram.



Compute the list of xy positions in the focal plane, store them in a list, display the list stored as long as there is no change required. If a change is required, recompute the list.

Roughly 200 microseconds are required for the housekeeping, bit-setting and storage into list per vector. Let the program be trapped for 30 microseconds (longest allowed vector), i.e. 60 ms for one vector. With 60 ms per vector, 417 vectors can be drawn. This assumes a list of vectors stored in proper form.

3. Comparison

For a static picture repeated computation allows a maximum of about 70 vectors while list display can process about 420 vectors. With list display it has been assumed that the program is trapped while vectors are drawn. If updating and rotations are necessary the number of vectors must be reduced. The rotational work is proportional to the total number of vectors. If .1 of the time must be reserved 63 vectors are possible with approach (1), while 378 vectors could be drawn otherwise. To avoid pulsation, however, list computation must be carried out over several periods (only 60 vectors can be computed and stored per period) before a new list is ready for display. It is not possible to display a partially updated list because of distortion.

Hence, the reduction in the number of vectors drawable is approximately twice as large with procedure III.B.2. Still 336 vectors can be drawn.

A remedy for case III.B.2 is possible by not requiring program trapping while vectors are drawn. In this case all updating and rotational computation can be done while drawing of vectors is accomplished on a cycle stealing basis. All 420 vectors can be drawn. Within slightly more than 7 periods a list is recomputable. Next the results of section III.B are tabulated. See Table I.

	Interrupt EOV, EOL	Program Trapped During Vector Drawing	Max. No. of Vectors Possible	Reduction in No. of Vectors Required if Rot. Comp.
Repeated Computation	No	No	70	$f \cdot n_v$
List Processing	Optional	Yes	420	$(2f) \cdot n_v$
List Processing	Yes	No	420	No Reduction

n_v = number of vectors in list

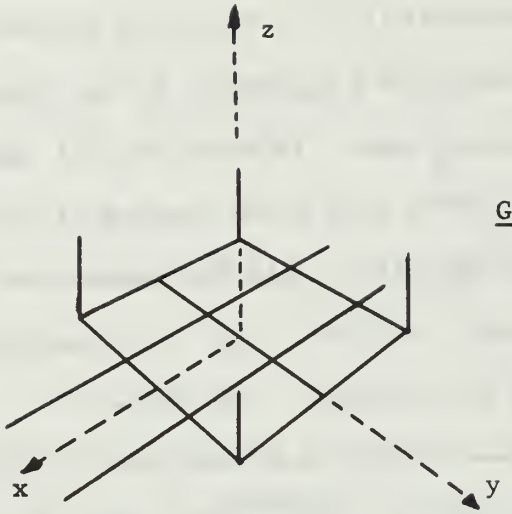
f = fraction of computation time required

TABLE I

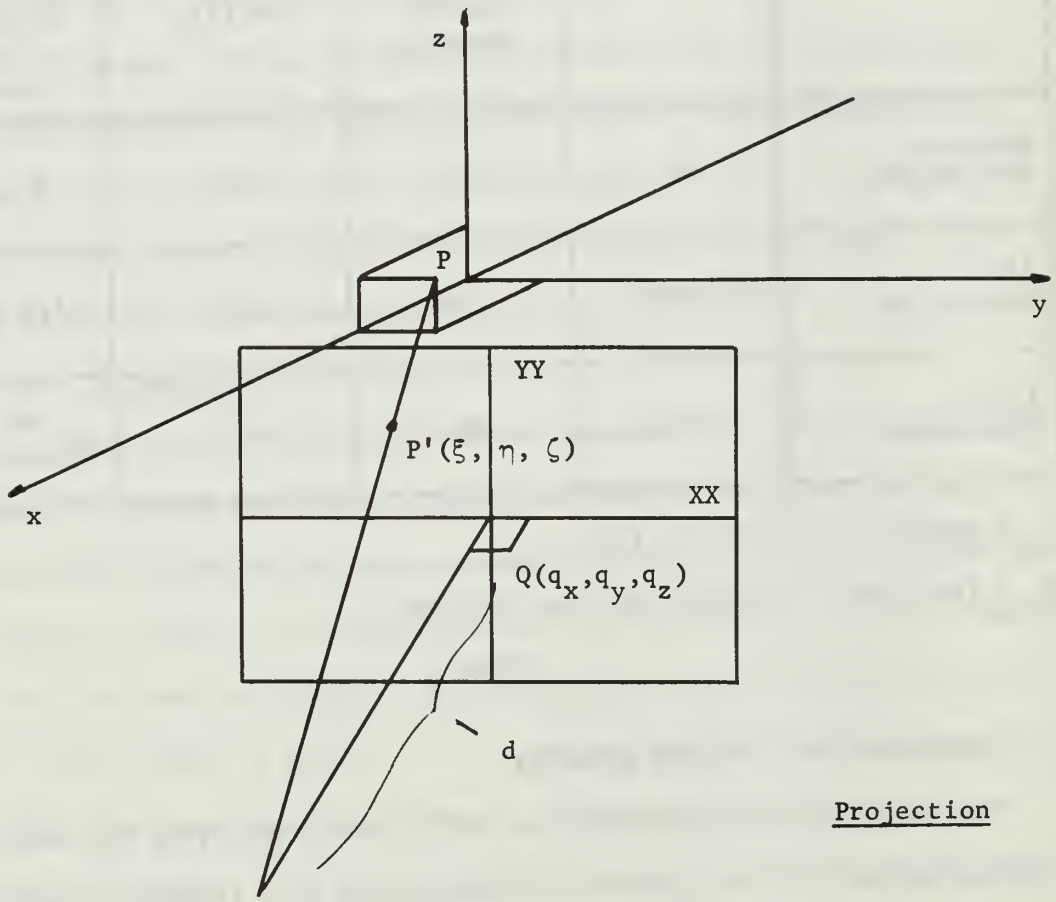
C. PROJECTION OF A CHOSEN GEOMETRY

The restrictions described above are silent directions for every programming effort that aims at implementation of a flight situation.

A simple geometric structure in three dimensions has been constructed and serves as the data the managing program will assign to various subroutines to operate on. See fig. 3.



Geometric Structure



Projection

FIGURE 3

The vector end points are sequentially transferred to a projecting routine that transforms three dimensional coordinates into planar coordinates of the focal plane. The routine that has been written is general in that it will project 3D coordinates onto the focal plane from any focal point selected. The distance of the focal point from the focal plane is allowed to vary. Also the focal vector may vary. Parameters required are:

C (c_x, c_y, c_z), the coordinates of the focal point;

alpha, angle of line of sight (focal vector) with the positive x-axis;

beta, angle of focal vector with y-axis;

gamma, angle of focal vector with z-axis;

d, distance between focal plane and observer.

The projection program (called CVTXY) implements the following equations

$$q_x = c_x + d \cos \alpha \qquad [d \equiv \text{DSTNC}]$$

$$q_y = c_y + d \cos \beta$$

$$q_z = c_z + d \cos \gamma$$

$$KK = d \Big| [(x-c_x) \cos \alpha + (y-c_y) \cos \beta + (c-c_z) \cos \gamma]$$

$$\xi = c_x + KK\ (x-c_x)$$

$$\eta = c_y + KK\ (y-c_y)$$

$$\zeta = c_z + KK\ (z-c_z)$$

$$XX = [(\xi-q_x) \cos \beta - (\eta-q_y) \cos \alpha] \Big| \sin \gamma$$

$$(1) = [-(\xi-q_x) \cos \gamma + (\zeta-q_z) \cos \alpha] \Big| \sin \beta$$

(for $\sin \gamma = 0$)

$$YY = (\zeta-q_z) \Big| \sin \gamma$$

$$(2) = (\eta-q_y) \Big| \sin \beta \quad (\text{for } \sin \gamma = 0)$$

Equations (1) and (2) have not been implemented.

See figure 3. First the coordinates of Q are found. Q is the point of intersection of the focal plane with its normal through C. Computed then are the coordinates of P' which is obtained by intersecting the ray from P to C with the foacl plane. ξ, η and ζ are then used to find the two-dimensional coordinates XX, YY in the focal plane.

(0, 0) in the focal plane is given by Q. For further information see ref. 13.

The projection program can be used by any user provided the following restrictions are met:

given that the focal vector points to the origin, the linear extension of the geometric structure may not lie beyond 25% of the total word length used (15 bits, maximum number = 37777_8, implies a maximum linear extension of 10000_8). If the extensions become bigger, overflow cannot be avoided. Overflow prevention operations have only partially been implemented to avoid time losses by a complicated computational structure. This structure will eventually lead to floating point arithmetic which had been excluded at the beginning.

D. ROTATION

The rotation program (called D1) implements rotations (orthogonal transformations) around the y-axis and around the z-axis according to the following equations

$$\left.\begin{aligned} \begin{pmatrix} x \\ y \\ z \end{pmatrix} &= [L] \begin{pmatrix} x' \\ y' \\ z' \end{pmatrix} \\ [L] &= \begin{pmatrix} u & -v & 0 \\ v & u & 0 \\ 0 & 0 & 1 \end{pmatrix} \end{aligned}\right\} \text{ for rotation around z axis}$$

$$= \begin{pmatrix} u & 0 & v \\ 0 & 1 & 0 \\ -v & 0 & u \end{pmatrix} \Bigg\} \text{ for rotation around the y-axis}$$

$$u = \cos\theta, \quad v = \sin\theta$$

$$\begin{pmatrix} x' \\ y' \\ z' \end{pmatrix} = \begin{pmatrix} u & v & 0 \\ -v & u & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix} \Bigg\} \text{ for z-axis rotation}$$

$$\begin{pmatrix} x' \\ y' \\ z' \end{pmatrix} = \begin{pmatrix} u & 0 & -u \\ 0 & 1 & 0 \\ v & 0 & u \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix} \Bigg\} \text{ for y-axis rotation}$$

For further information on orthogonal transormations, see ref. 7.

Rotation around a general set of coordinate axes, defined implicitly by specifying c_x, c_y, c_z, alpha, beta, gamma, is demonstrated in a program called ROT.

E. APPLICATION TO FLIGHT VEHICLE MISSION PERFORMANCE EVALUATION

Projection and rotation are the basis of any perspective picture usage in performance evaluation. Since the programs are general from a mathematical point of view, they can be used to display any function of two variables. The part of the function to be projected must be broken up into a series of short vectors, depending in length the frequency content of the function. To have a satisfactory display of a three dimensional structure, it is in general necessary to solve the hidden line problem. This problem has not been tackled in this project for the following reasons:

(a) The time requirement of a general hidden line suppressing routine will be beyond what can be tolerated in an application where refreshing must be done by the CPU.

(b) A simple geometrical structure of limited extension viewed from a large distance will not be destroyed by anticues, if the variations in the z-direction are few and isolated. It follows that the routines provided in this project are not general enough to be used in connection with, say, engineering drawings which shall be viewed from all sides. If there is no requirement for a dynamic display, then a hidden line suppressing routine can be added to the existing routines. For the purpose of demonstrating the capability of a digitally controlled CRT in flight vehicle performance evaluation, they are not required. The following part of a flight simulation has been implemented (the pilot is imagined to be at the focal point):

Let the structure of the geometry be given as shown in fig. 3.

Let it correspond to an airfield (one can think of a real display with a PSEUDO 3D). Upon approaching this field the eye of the pilot (he is allowed unshaded view into all directions) will sense the changed perspective with respect to the airfield as a result of his motion. Proceeding towards the airfield on the same flight level will change the perspective to a top view. Side translation will result in a rotation around the z-axis. The presentation of the field is permanent which means that the observer's eye is fixed to the destination and not following the course of the vehicle. It is a matter of a side translation to move the structure out of the scope, if it is desired to have the operator's eye restricted to a small viewing angle (solid angle) in the direction of the course of the vehicle. A program has been provided that limits the x and y values when they are about to overflow (see Appendix). It has been demonstrated, that a permanent perspective display of the bearing of the destination has its attractions.

The sequence of perspective pictures that will result from a moving vehicle can be obtained in a computer application in two different ways:

(1) One end of the focal vector is fixed on the object, the position in space to which the other end is connected is changing, hence the focal plane will follow a rotation and translation. $C(c_x, c_y, c_z)$, alpha, beta, and gamma must be updated permanently. A straightforward solution will result. The computational work necessary to perform updating is lengthy and difficult to carry out in fixed point arithmetic. It requires evaluation of three dot products and evaluation of inverse cosines.

(2) Instead of letting the aircraft move around the airfield, let the relative motion of the airfield around the aircraft be used as a basis for computing the perspective change of the airfield. Instead of allowing the airfield to move and being repositioned by coordinate transformation (translation), which is necessary in fixed point arithmetic, leave the airfield permanently as the origin of the focal vector. Use the computed change of perspective to rotate the geometrical structure.

This simplification is only justified under certain conditions:

(1) extensions of the geometric structure in view are small and confined to a certain region,

(2) the (initial) c_z must be large compared to the highest objects in the structure.

There are difficulties in providing magnification as a measure of changing distance encountered the solution of which is so far unsatifactory.

Under the assumption that a satisfactory solution to the last problem can be found, the method has attractions:

(1) considerably less computational work, simple implementation, only one focal point and one value for each of C (c_x, c_y, c_z), alpha, beta, gamma is required; part of the projection program is repetitive and can be bypassed,

(2) no coordinate translations are necessary.

Beyond the advantages stated, the speed of the operation can later be improved by attaching hardware which can accomplish rotations.

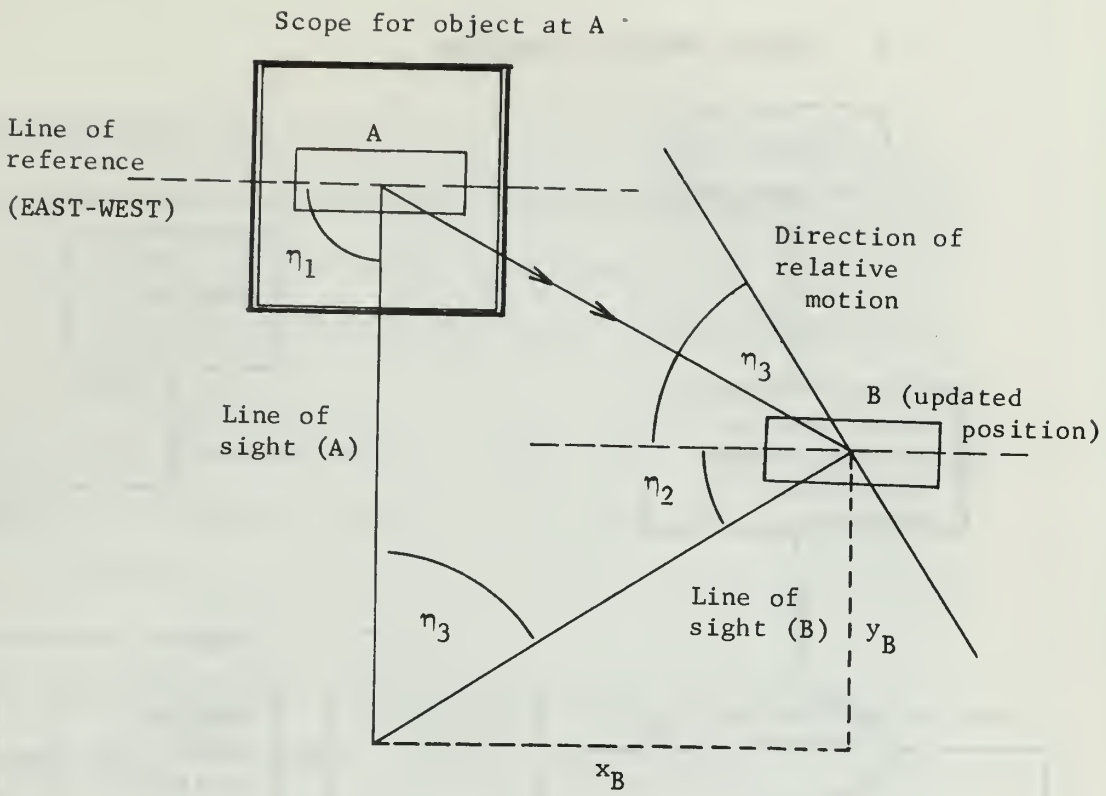
In order to achieve the visual impression of an approach to an airfield, it is necessary to control the rotation of the structure to be projected according to the computed perspective changes. The simplest way to control rotations is to project along the absolute x-axis and rotate the geometry by the proper value initially for true geographical reference. From then on altering the vehicle's course and speed will have consistent effects on the perspective.

The angles by which rotation must be carried out are obtained by a simple simulation routine. According to the motion in the xy-plane, the updated position is used to compute the angular change with respect to a reference line. See fig. 4.

For a relative motion of the object from A to B the angle by which the object appears to be rotated is given by η_3,

$$\eta_3 = \pi/2 - \arctan (y_B/x_B).$$

When the managing routine calls the update routine, the value for the rotation is computed and stored. Before the vectors are displayed, an investigation is made if there is a need of an angular or horizontal rotation. If so, rotation is carried out before projection.


Scope for object at A
Line of reference (EAST-WEST)
A
η_1
Direction of relative motion
η_3
B (updated position)
Line of sight (A)
η_2
η_3
Line of sight (B)
y_B
x_B

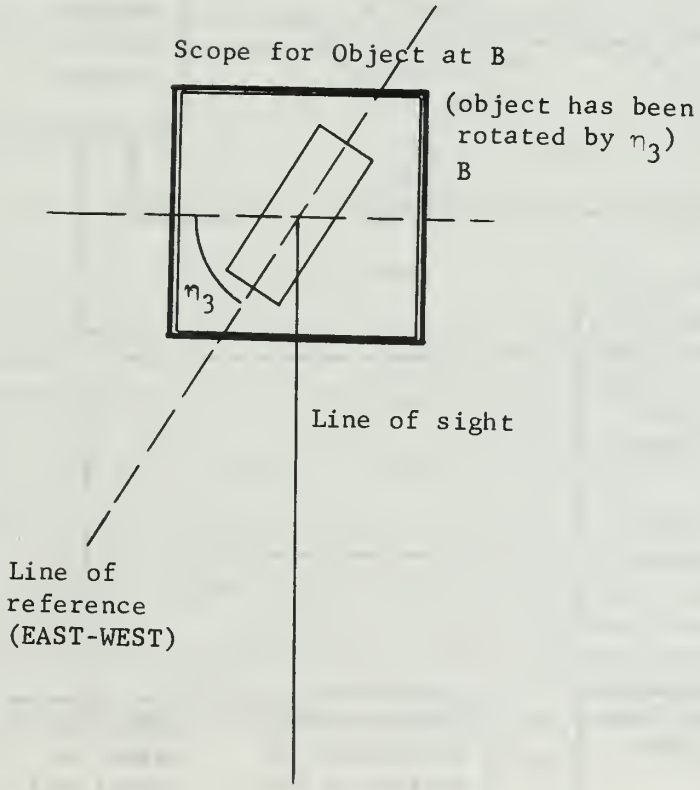

Scope for Object at B
(object has been rotated by η_3)
B
η_3
Line of sight
Line of reference (EAST-WEST)

FIGURE 4

F. OVERALL PROGRAM STRUCTURE

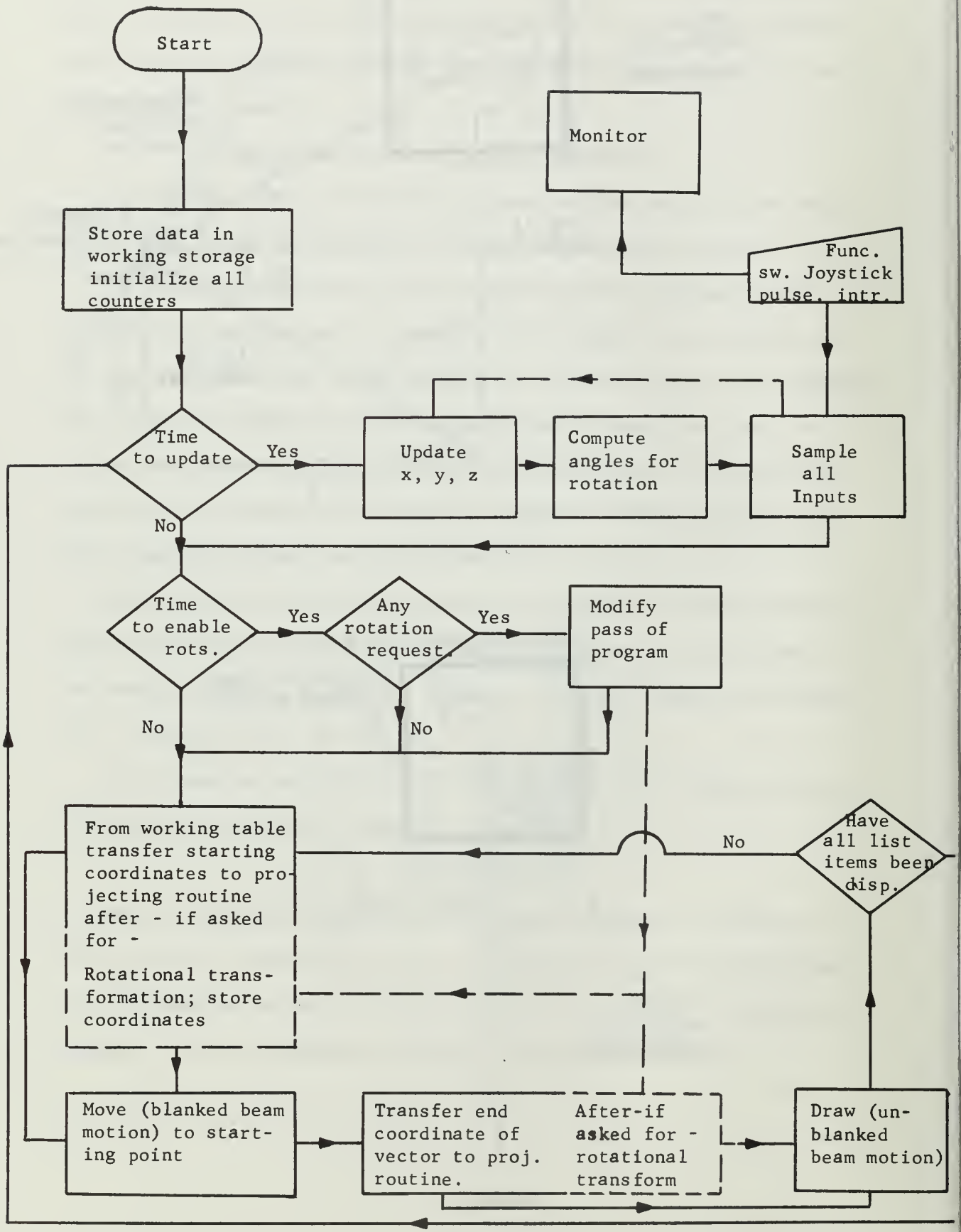


FIGURE 5

G. CRITICAL COMMENTS

The overall managing program, given in block diagram form (fig. 5) can be regarded successful in the following aspects:

(1) implementation of projection rotation and translation in a running process,

(2) vector generation from the digital processor (DPR2) instead of using the display operator provided by the Adage Company,

(3) effective demonstration of a section of a flight situation (see photographic sequence),

(4) an initial step in the direction of limiting has been carried out (see Appendix C).

The overall managing program leaves several aspects open for further studies:

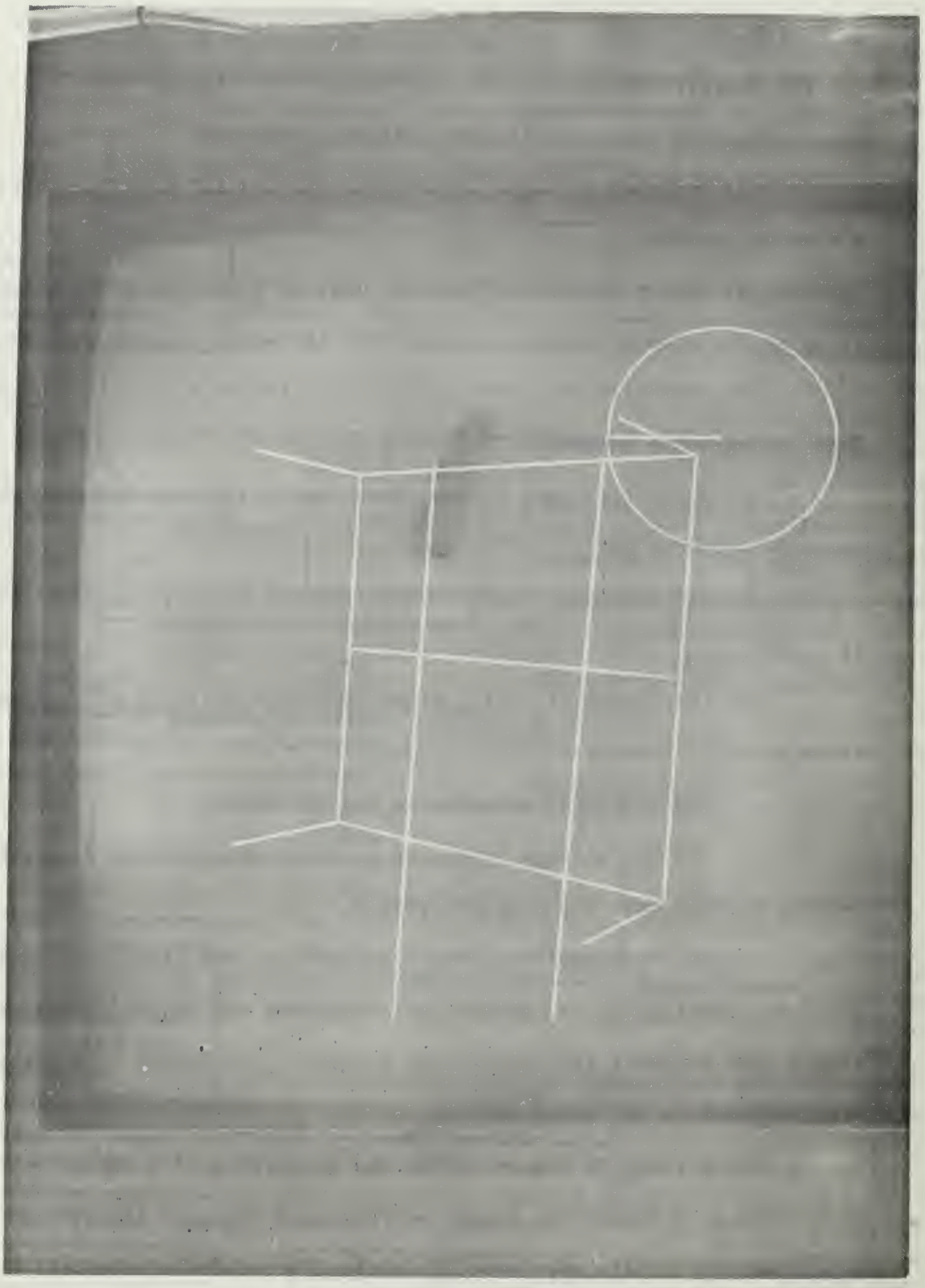
(1) the flight situation demonstrated is still in an experimental state -

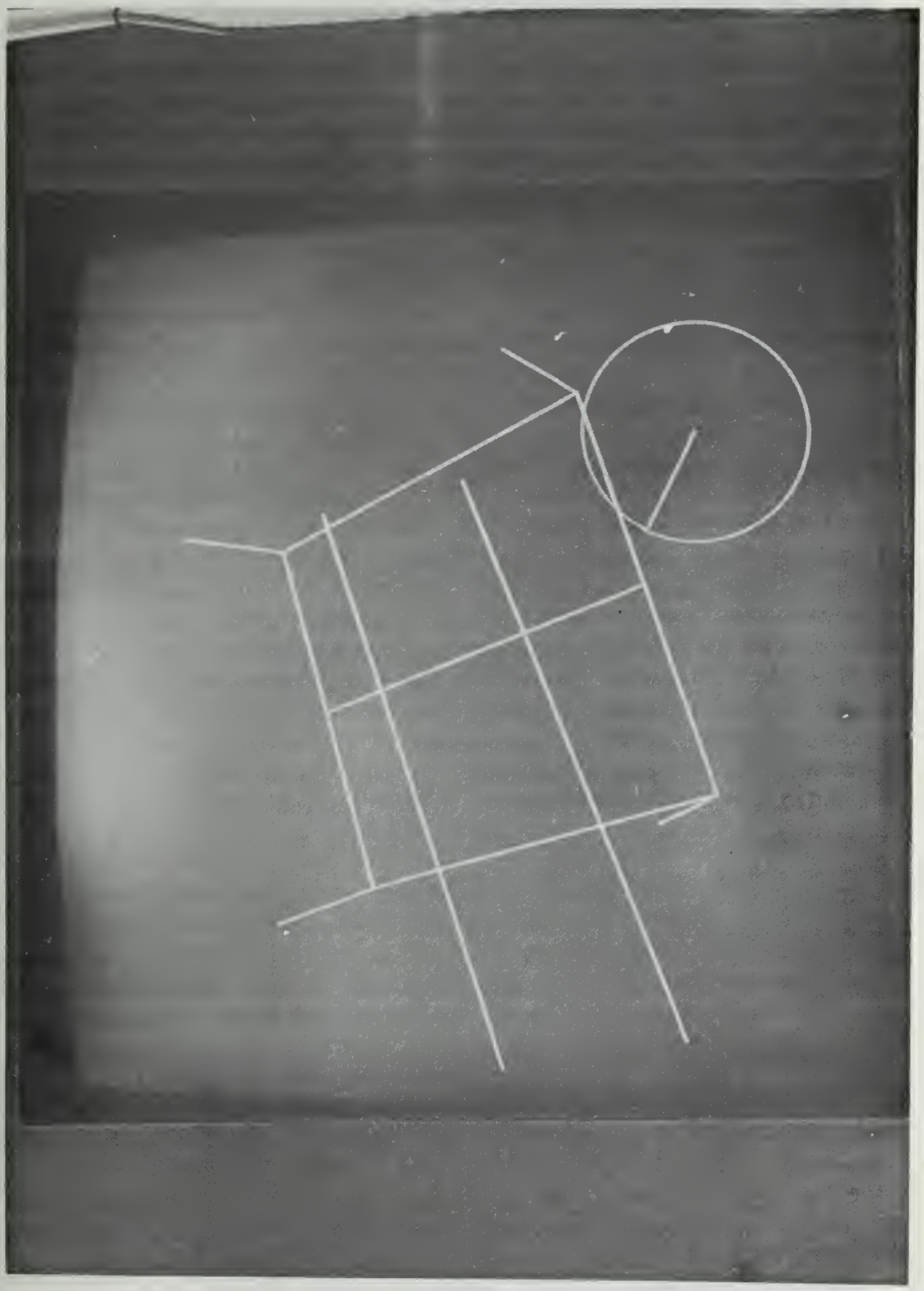
(a) timing of managing is not optimized,

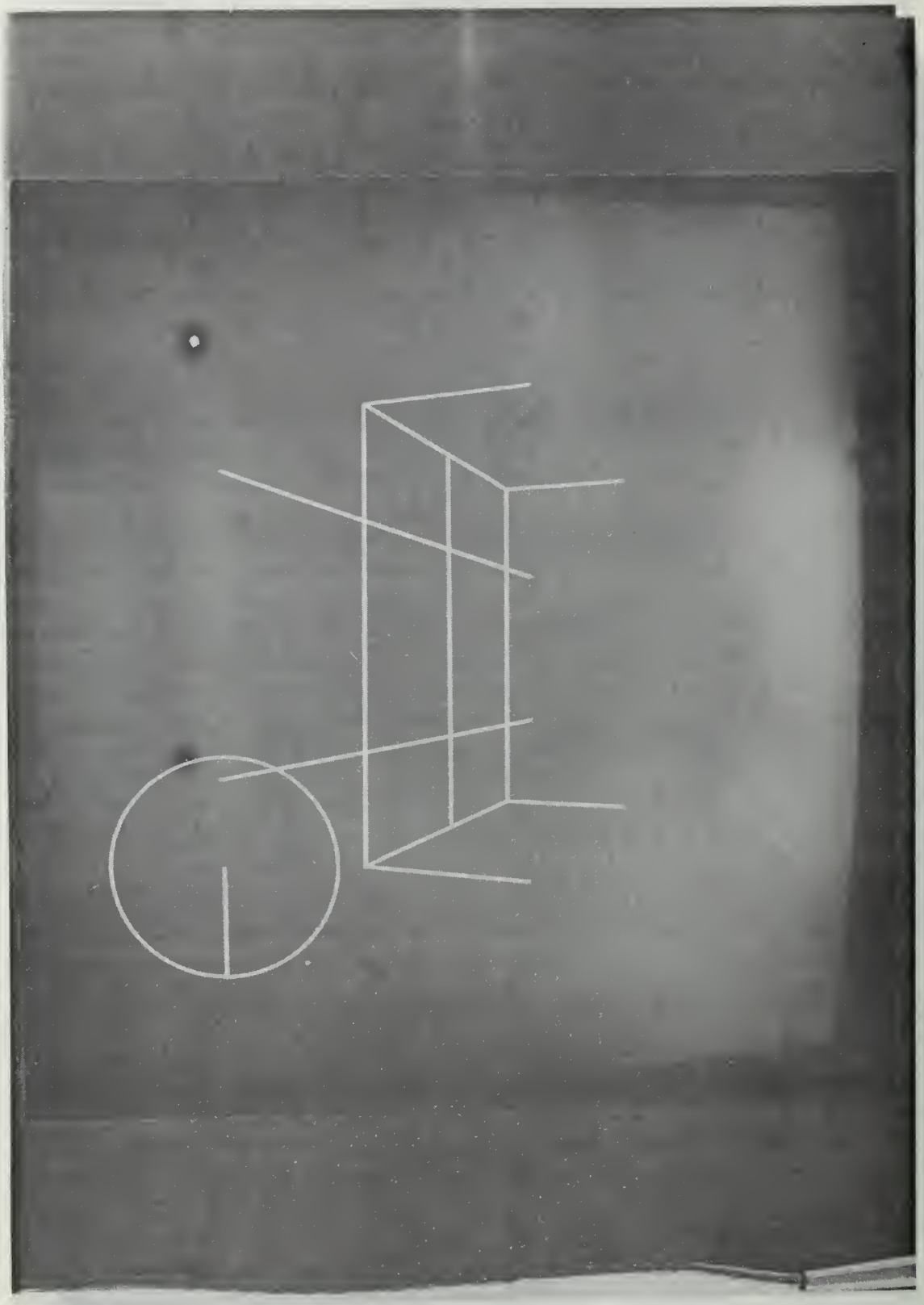
(b) the minimum levels of rotation that provide a smoothly varying display have not been determined,

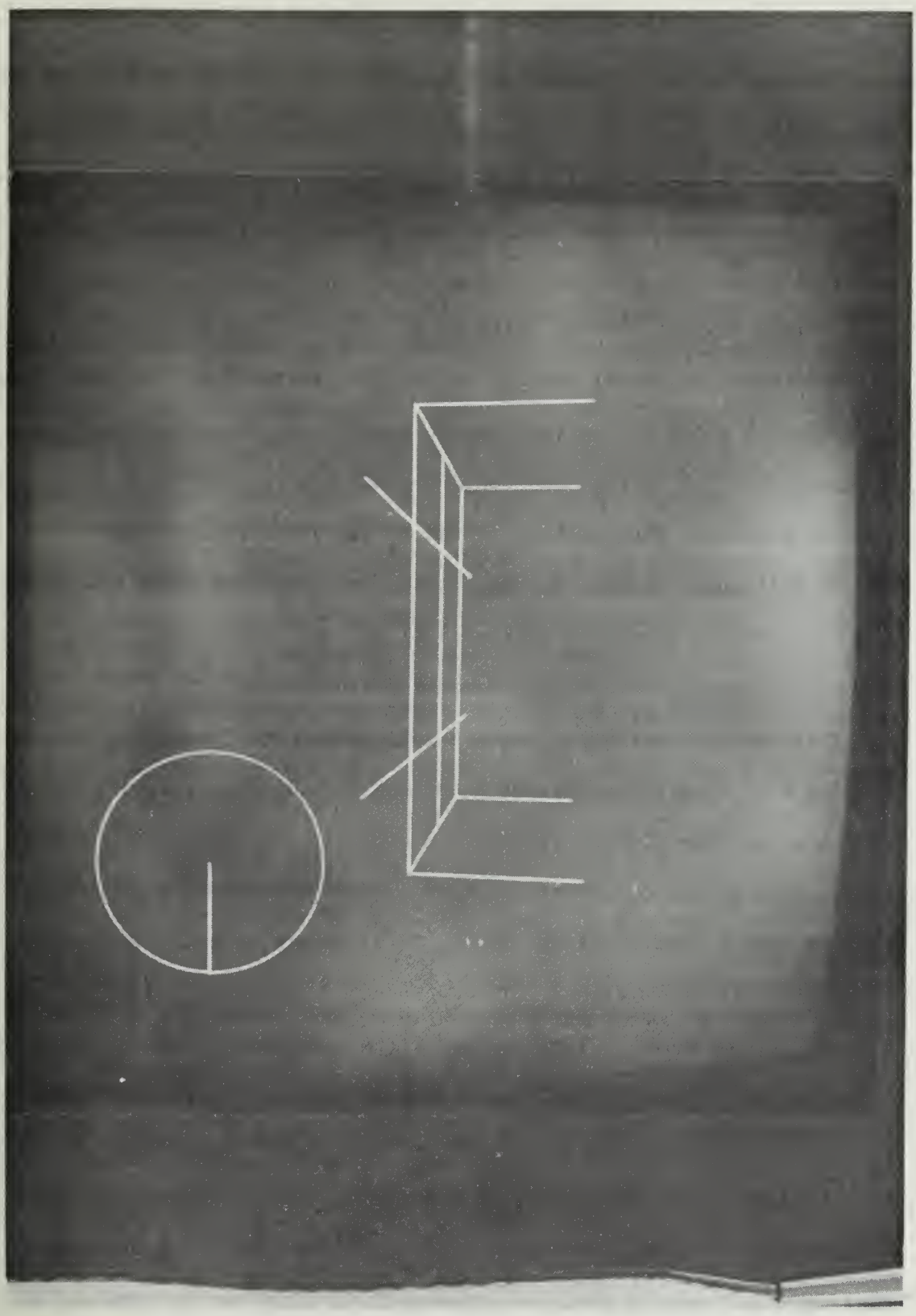
(c) the aspects of final approach to the 'air field' have not been investigated, a routine that magnifies the object proportionally to the distance and reduces the geometric extensions by software limiting need to be introduced,

(2) a study is suggested for the generation of a set of vectors describing in a simple but proper way a chosen 'target' area, which must be allowed to contain any three dimensional structure whatsoever.









H. PROPOSALS FOR APPLICATION OF THE TECHNIQUE

Without further investigations an assessment concerning the general applicability of the technique in an arbitrary flight situation cannot be made. This is especially true with the problem of finding a set of vectors describing the geometry being unsolved. But there are certain cases in which the destination geometry is very simple; in a ship to ship guided missile problem there is no difficulty in defining the target geometry. A lack of profile might be a problem here. A combination of PSEUDO 3D elements and real target description is no problem from a programming point of view and could be employed to ease the guiding problem greatly. When looking at the photographic results there is no way to tell which vectors belong to the real world and which are superimposed. Once a geometric reference is set up a misinterpretation is impossible when geometric measurements are updated.

There is one particular feature that should draw interest: by showing the perspective view of an object as seen from a moving observation point (the view is no function of roll, pitch and yaw!) all the time, it is possible to train personnel in minimum time to bring a vehicle into a desired target area. This feature is effectively demonstrated (see photographs). The sequence of photos mentioned depicts a situation as shown in fig. 4., where the technique is described, that leads to the sequence. The photos show also the true course of the vehicle and its radar picture, superimposed in the same display. Without familiarization with the significance of the different types of information combined, the display will be confusing. This suggests that certain types of familiar presentations might not be combined at all. They are shown here to demonstrate the dynamic capability of the

display, and to allow control of the perspective rotations in the frame of the experiment.

IV. CONCLUSIONS

A programming project has been carried out to

- study aspects of flight vehicle performance evaluation with the help of a digital CRT,

- investigate within the realm of the application the capability of the AGT10 to perform the assigned task.

A simple technique for operation (or guiding) a flight vehicle has been described and implemented in an experimental form. Some possible fields of application have been pointed out, also proposals for further studies have been made. The capability of the machine to perform a certain amount of computational work in updating and sampling controls has been demonstrated. With increasing complexity of the geometric structure list processing has been found the way that leaves a maximum of computation time available for other work. Updating and refreshing in one loop will be below the rate required as soon as the number of vectors to be displayed becomes larger. Suggestions have been made to introduce hardware in the rotational work, this will allow for more computation time in a tight situation.

Fixed point arithmetic must be employed if the refreshing rate necessary for flickerless display shall be maintained. With fixed point arithmetic there will always be a finite probability of overflow if numerical restrictions are not met.

APPENDIX A

VECTOR GENERATION FROM THE DIGITAL PROCESSOR

There exists a program package called DSPLY, the so-called display operator, provided by the Adage Company. This system of programs furnishes means of drawing vectors, points and simple structures the defining coordinates of which must be described in machine language format. The aid consists of providing statements which are resolved into sequences of machine language instructions, which guarantee proper timing in transferring values to the various registers involved in a vector drawing operation. By avoiding these problems some advantages are lost and new problems are created (large storage requirement because of its generality). Much understanding for the otherwise often meaningless formats of the display operator is gained by studying the basic steps of vector generation directly from the DPR2. A dynamic display, however, requires the most efficient vector generation and requires more storage than the display operator leaves for use.

The following registers (destinations) must be provided with a value to control every phase and aspect of vector drawing:

Destination 10 (D10), upper 14 bits

destination 11 (D11), upper 15 bits

destination 7 (D7), both half-words

destination 6 (D6), lower 15 bits

destination 5 (D5), both half-words.

The sequence shown above is mandatory. If a value is already set, its repetition can be omitted, except the transfer to destination 5. The

most important registers are destination 10 and destination 5. The operating modes of the vector generator are controlled by the values in the upper 14 bits of destination 10. D5 is the destination register for the XY pair to describe one end of a vector. One can distinguish two major operating modes of the vector generation: flag bit control and mode bit control.

M4	M3	M2	M1	DASH	DCL	B1	B2	B3	B4	P1	P2	P3	P4	XXX
0	1	2	3	4	5	6	7	8	9	10	11	12	13	

Control word for destination 10

A. FLAG BIT CONTROL

M4 = 1 directs the vector generator (AVG) to investigate bits 14 and 29, if a vector shall be drawn (bit 29 = 1) or if a vector shall be moved (blanked motion; bit 29 = 0). A vector drawing requires a MOVE operation to the starting point and a DRAW operation to the vector end point. If the starting point is the end point of another vector, MOVE can be omitted. Bit 14 = 1 signals end of list (EOL), in which case the AVG is switched off after the last vector drawing. Various options exist under flag bit control mode: the details can be obtained from the "Subsystem Programming Specifications" for the AVG. In the following is described, how a list is most suitably processed in a majority of applications:

If M3 = 1, at the end of a vector drawn an 'end of vector' (EOV) interrupt occurs and control is transferred automatically to pivot location 77756. The instruction

MDIR'X 77756

is executed. The instruction to be taken to the instruction register (IR) is the one stored in 77756. If previously

MD05 DATA

(transfer from memory location DATA to destination 5) has been stored in 77756, then the next instruction is transferred to destination 5 the effective address being now DATA + 1 - due to the indexing operation which incremented the contents of 77756. DATA is the starting address of the list of values that represent the beginning and the end of all vectors to be drawn. DATA might typically be a set of vectors to construct a subpicture. The transfer to D5 is stopped if a 1 bit in position 14 is discovered signalling the end of the list. No option exists to rule out the end of list interrupt under flag bit control.

When the interrupt occurs control is transferred to pivot location 77757 and the instruction

JPSR'I 77757

is carried out. Whatever address is stored in 77757, the next instruction is taken from this address plus 1 in accordance with the correct interpretation of JPSR. The interrupt Flip Flop remains set until a JUMP'I instruction is encountered which clears the interrupt. If the address stored in 77757 is another transfer sequence D10, D11, ..., D5, the basic refreshing loop is set up. With the help of an end of list interrupt it is possible to sequence several independent lists of variable length. All lists together might form a picture of a complex structure.

Transfer of the list values is made on a cycle stealing basis. If all list values must be modified before displaying can start (e.g. rotation of a body), the basic refreshing loop as described above can easily be altered.

The following instructions set the proper bits to accomplish a vector generation from a list with EOV and EOL interrupts (also all other register values are included) and trigger processing:

```
MD1Ø'0          ADDR1          /operating mode of AVG
MD11            ADDR2          /scale
MDØ7            ADDR3          /offset
MDØ6            ADDR4          /intensity
MDAR            ADDR5          /store next list addr in
                                77757
ARMD            77757          /store next list addr in
                                77757
MDAR            ADDR6          /store MD05 DATA in 77756
ARMD            77756          /store MD05 DATA in 77756
MDØ5            DATA           /transfer first XY pair
continue
```

In locations ADDR1, ADDR2, ..., ADDR6 the following constants are stored:

```
ADDR1   :   6Ø4ØØ   !   H
ADDR2   :   37777   !   H
ADDR3   :   Ø
ADDR4   :   2ØØØØ
ADDR5   :   Ø       !   H TABLE
ADDR6   :   MDØ5    !   H DATA
DATA    :   X1      !   H Y1
            .           .
            .           .
            .           .
```

B. MODE BIT CONTROL

Mode bit control is exerted whenever M4 = 0. Bits (14) and (29) are of no effect. A variety of options exists. One very useful for individual MOVE or DRAW operations is outlined here:

No interrupts occur at the end of the vector drawing operation. Hence the program must assume what the vector lengths and the resultant drawing times will be. The interval between any two transfers to destination 10, 6, or 5 should be about 8 microseconds (the equivalent of 2 normal instructions). A transfer too early acts like an inhibit operation and no vector at all is drawn. Provided the other registers have been assigned the desired value previously, the instruction

MD1Ø'0 LOC1

will initiate a MOVE operation, where LOC1 : 1Ø4ØØ ! H

and the instruction

MD1Ø'0 LOC2

will initiate a DRAW operation, where LOC2 : 144ØØ ! H.

Obviously a word must have been transferred to destination 5 between the above two instructions and a transfer is expected after the latter one.

With the help of the AVG subsystem programming specifications it is easy to resolve the word at location LOC2 into its information content: Bits 2, 3, and 6 are set, i.e. M2 = M1 = 1, M4 = M3 = 0. M4 = 0 implies mode bit control, M1 = 1 implies draw a vector in two dimensions, bit 6 = 1 implies scope I on. To utilize the various options provided, it is easy to make up the proper word. When changing from a MOVE to a DRAW and back to a MOVE it is necessary to interpose an AND operation to make sure that the proper bit (M1 in this case) is on or off

respectively. Otherwise all MOVE instructions are interpreted as DRAW instructions.

C. SAMPLING OF FUNCTION SWITCHES AND JOYSTICK

1. Function Switches

Very often joystick and function switches are used in connection with vector drawing operations. The function switches individually have a value of 1 as long as they are depressed only. If IC (9) (interface control register) is on, the upper 15 bits of source 5 are connected to function switches 1 to 15. An instruction sequence

```
        MDIC'0'L
        4Ø  !  H
        S5MD              FSW
FSW  :  Ø
```

will initiate the transfer of the instantaneous state of S5 to the memory location FSW. With suitable AND operations the various bits can then be investigated. If IC (9) = 0, then the states of function switch 16 and the foot pedals are connected to S5. Obviously the contents of source 5 can be transferred to any legal destination like AR, BR, ...

2. Joystick

A different procedure must be followed if it is desired to obtain the instantaneous value of the joystick potentiometers. There is no analog to digital converter available with the AGT10. Instead a digital comparison value is converted to an analog one and a comparison is made. Source 4 bit 24 is on or off, depending on whether or not the difference between the comparison value and the unknown quantity is greater or less than zero. The comparison value is sent to destination 7, bits

0 to 14. A multiplexing operation (initiated by switching on bits 22, 23, or 24 of destination 11) selects the desired variable for comparison. It is possible to compare the sum of two unknown voltages. A sequence of yes or no questions (10) leads to a digital value of 10 bits. When the proper voltage value has been found, it is normalized with the maximum number found at the joystick. A program called 'HEBEL' can be called. The sampled value is returned in the lower 15 bits of the A-register. When entering the subroutine, the A-register must contain the values used in multiplexing (20 for the x-voltage, 40 for the y-voltage). When the end values of the potentiometers are not set to their proper value, normalization will give a meaningless result. A check of proper adjustment can be made by assembling a program called 'ATODE', pushing at execution joystick into any extreme position. A listing of the values in locations ATODE + 16_8 and ATODE + 17_8 should show the corresponding values. Programs HEBEL and ATODE are listed below.

APPENDIX B

PROGRAMMING HINTS FOR ADAGE ASSEMBLY LANGUAGE

Careful study of the programming instruction manual provided by the Adage Company will not provide adequate preparation for several minor difficulties. The following remarks are intended to save experimentation time when first working with the terminal.

(1) Multiplication involves the extended arithmetic unit (EAU1). It uses more time than a normal instruction. Hence, if the instruction that succeeds uses the A-registor, too, a NOOP must immediately follow the multiply instruction.

The same holds for the divide and shift instructions. The latter ones require a NOOP, however, only in the case of the shift being more than a half-word.

(2) MPYU LOC multiplies the lower 15 bits of the accumulator by the contents of the upper half of the word in location LOC; MPYL LOC multiplies by the lower half of the word in location LOC, assumes bits 1 to 14 equal to zero, but taking part in the multiplication operation.

(3) Local storage assignment when programming in assembly language: can be done with the help of a macro --

```
REPEAT    ...
          0
ENDR
```

The number replacing the three dots must be octal.

(4) Operating recommendations for AGT 10:

Program execution. To execute a program on the AGT 10 it is necessary to load the program from magnetic tape first. A teletype statement to the monitor START ("program name", 0)! will direct the monitor to load the program from tape with the help of the loader. The tape unit is indicated by "Ø". All external programs needed are loaded and linked, too. A statement 120 ! or GOTO (120) ! will execute the program loaded. 120 can be replaced by any number consituting a meaningful entry point. For further details see programming specifications for AMRMX (name of system). A load and go procedure is suitable for some programs which is initiated by 'program name'!

Generation of a program using assembly language. At the end of an assembly, the object program is appended as the next file on the user's tape. The assembler will operate on the source language statements in the scratch pad (file 10, first 24 feet of tape immediately following the load point) of the user's tape. The following steps are necessary in the production of an object program:

(a) write source program on the scratch pad,

(b) load assembler from mag tape and execute assembler (either START ("ADEPT", 0) and ADEPT!, or simply ADEPT! as a load and go).

To ( ): To write text onto the scratch pad is done with the help of a program called EDIT. When EDIT is loaded and executed the statements written on the scratch pad can be controlled from teletype. A variety of instructions exist (see Manual on EDIT). After text 'edition' has been completed, a control statement 'B, return' will try to return control to the monitor (system). Since EDIT overlays part of the system, however, it is necessary to reload the system from tape. This is done

from the control panel (OPC) with depression of the following sequence of buttons:

HALT - RESET - RUN - clear buffer register and set 1001_2 - PULSE 1. Not earlier than after the depression of PULSE 1 the loader (which is not overlayed by EDIT under legal circumstances) will begin a tape search for file 11, which contains the system. When file 11 is found, a teletype statement MO/DA/YR = is issued; it must be completed properly.

Step (b) can now follow.

To (b): At the end of the assembly, it is again necessary to reload the system from tape, since the storage requirement of ADEPT and further necessary programs are such that the system is overlayed. If by some unwanted or unexplainable action the loader has been destroyed, it must be reloaded from paper tape (Bootstrap). An indication of a destroyed loader is often tape motion before PULSE 1 has been depressed when trying to load from tape. The procedure then is: depress buttons RESET and BTSTR in that sequence, then start paper tape reader.

APPENDIX C

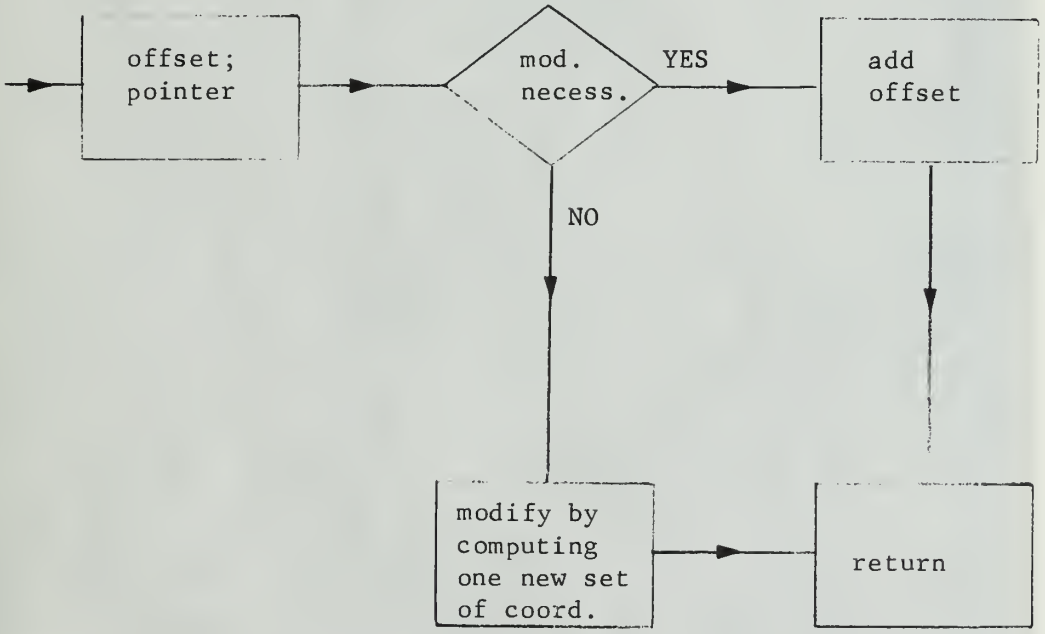
PROGRAM LISTINGS AND SHORT DOCUMENTATION

A. SUBROUTINE SCHB

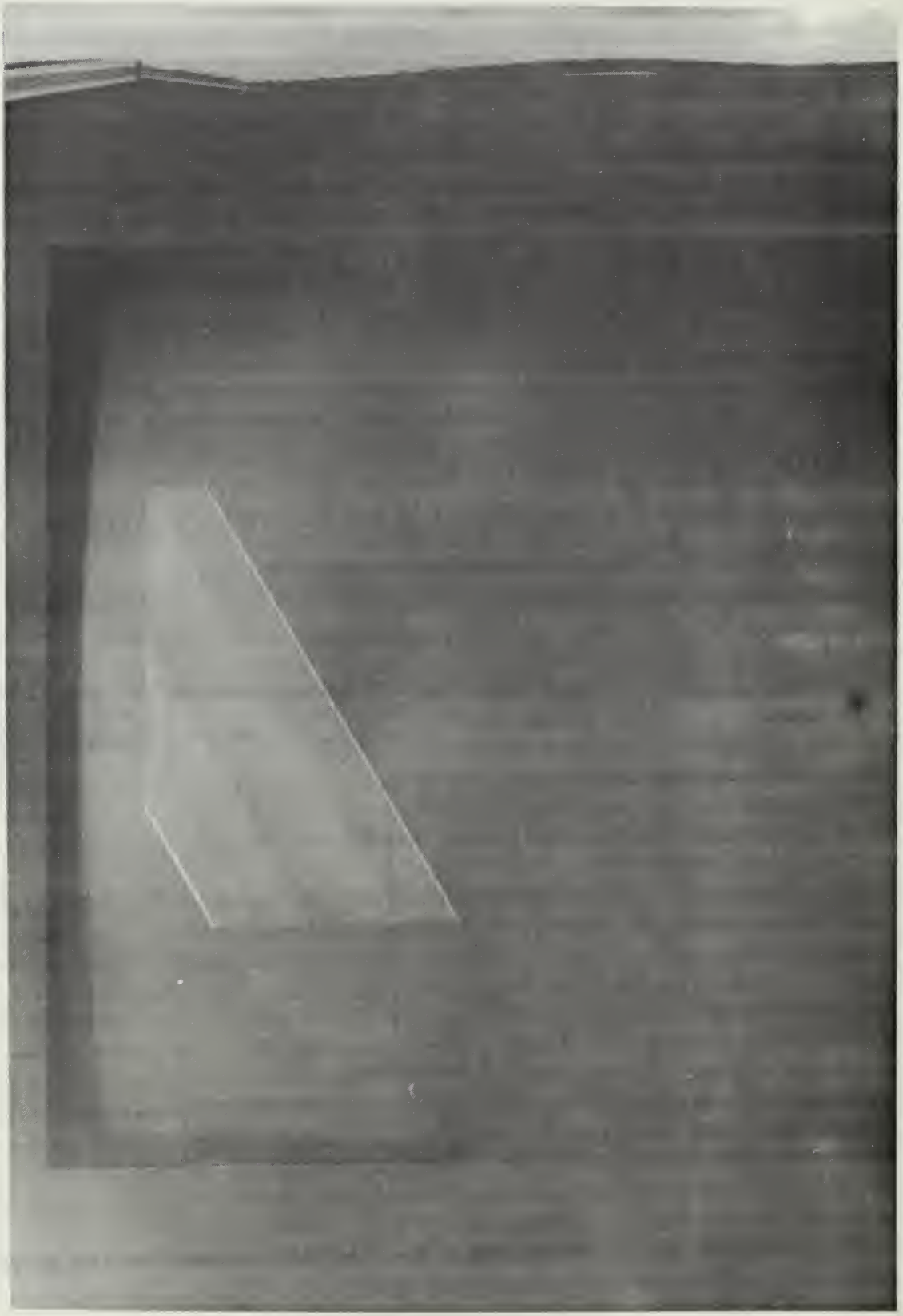
Implements a translation of a vector in one direction (y-direction). (If the x-direction is direction of translation, rotation by 90^{o} is necessary.) When a limit is met, scissoring is performed by computing a new vector end point.

The value by which translation shall be made must be stored in the accumulator before entering SCHB.

SCH expects the first address of the four vector coordinates (XX1, YY1, XX2, YY2) in the buffer register (Pointer).



See photograph on following page. Routine VSRS is used for the purpose of demonstration.



B. SUBROUTINE ROT

Computes rotational change of coordinates for rotations by $\pm 5^{o}$ around the x, y and z axis. Six entry points can be referenced to externally. The mathematical basis is shown in Chapter III. Input parameters are (common storage) PX, PY, PZ. New values are returned in the same locations.

C. SUBROUTINE HEBEL

Subroutine HEBEL has been described in Appendix A (joystick).

D. SUBROUTINE CVTXY AND D1

Subroutines CVTXY and D1 have been described in Chapter III.

```
EXPUNGE
TITLE ROT
ENTRY ROT,DRLFT,DRGHT,DRXPS,DRXNG,DRYPS,DRYNG
COMMON CX,CY,CZ,PX,PY,PZ,XX,YY,DSTNC,ALPHA,BETA,GAMMA,SBETA
COMMON F1,F2,THETA
ROT:    NOOP
D1:     NOOP
        MDAR'L
        00707!H
        ARMD            DTHETA
        JPSR            $COSA
        ARMD            CTHETA
        MDAR            DTHETA
        JPSR            $SINA
        ARMD            STHETA
        MDAR'N          DTHETA
        JPSR            $COSA
        ARMD            CFI
        MDAR'N          DTHETA
        JPSR            $SINA
        ARMD            SFI
        MDAR'L
        NOOP
        ARMD            DRLFT+1
        ARMD            DRGHT+1
        JUMP'I          D1
D2:     NOOP
        MDAR'L
        00707!H
        ARMD            DMEGA
        JPSR            $COSA
        ARMD            CEPS
        MDAR            DMEGA
        JPSR            $SINA
        ARMD            SEPS
        MDAR'N          DMEGA
        JPSR            $COSA
        ARMD            CKAPPA
        MDAR'N          DMEGA
        JPSR            $SINA
        ARMD            SKAPPA
        MDAR'L
        NOOP
        ARMD            DRXPS+1
        ARMD            DRXNG+1
        JUMP'I          D2
D3:     NOOP
        MDAR'L
        00707!H
        ARMD            DLBDA
        JPSR            $COSA
        ARMD            CMU
        MDAR            DLBDA
        JPSR            $SINA
        ARMD            SMU
        MDAR'N          DLBDA
```

```
        JPSR            $COSA
        ARMD            CNU
        MDAR'N          DLBDA
        JPSR            $SINA
        ARMD            SNU
        MDAR'L
        NOOP
        ARMD            DRYPS+1
        ARMD            DRYNG+1
        JUMP'I          D3
DRLFT:  NOOP
        JPSR            D1
        MDAR            PX
        MPYU            CTHETA
        NOOP
        ARMD            PR1
        MDAR            PX
        MPYU            STHETA
        NOOP
        ARMD'N          PR4
        MDAR            PY
        MPYU            STHETA
        NOOP
        MDAE            PR1
        ARMD'H          PX
        MDAR            PY
        MPYU            CTHETA
        NOOP
        MDAE            PR4
        ARMD'H          PY
        JUMP'I          DRLFT       [COMPUTED XPRIME,YPRIME
DRGHT:  NOOP
        JPSR            D1
        MDAR            PX
        MPYU            CFI
        NOOP
        ARMD            PR1
        MDAR            PX
        MPYU            SFI
        NOOP
        ARMD'N          PR4
        MDAR            PY
        MPYU            SFI
        NOOP
        MDAE            PR1
        ARMD'H          PX
        MDAR            PY
        MPYU            CFI
        NOOP
        MDAE            PR4
        ARMD'H          PY
        JUMP'I          DRGHT
DRXPS:  NOOP
        JPSR            D2
        MDAR            PY
        MPYU            CEPS
```

```
        NOOP
        ARMD            PR1
        MDAR            PY
        MPYU            SEPS
        NOOP
        ARMD'N          PR4
        MDAR            PZ
        MPYU            SEPS
        NOOP
        MDAE            PR1
        ARMD'H          PY
        MDAR            PZ
        MPYU            CEPS
        NOOP
        MDAE            PR4
        ARMD'H          PZ
        JUMP'I          DRXPS

DRXNG:  NOOP
        JPSR            D2
        MDAR            PY
        MPYU            CKAPPA
        NOOP
        ARMD            PR1
        MDAR            PY
        MPYU            SKAPPA
        NOOP
        ARMD'N          PR4
        MDAR            PZ
        MPYU            SKAPPA
        NOOP
        MDAE            PR1
        ARMD'H          PY
        MDAR            PZ
        MPYU            CKAPPA
        NOOP
        MDAE            PR4
        ARMD'H          PZ
        JUMP'I          DRXNG
DRYPS:  NOOP
        JPSR            D3
        MDAR            PX
        MPYU            CMU
        NOOP
        ARMD            PR1
        MDAR            PX
        MPYU            SMU
        NOOP
        ARMD            PR4
        MDAR'N          PZ
        MPYU            SMU
        NOOP
        MDAE            PR1
        ARMD'H          PX
        MDAR            PZ
        MPYU            CMU
```

```
          NOOP
          MDAE          PR4
          ARMD'H        PZ
          JUMP'I        DRYPS
DRYNG:    NOOP
          JPSR          D3
          MDAR          PX
          MPYU          CNU
          NOOP
          ARMD          PR1
          MDAR          PX
          MPYU          SNU
          NOOP
          ARMD          PR4
          MDAR'N        PZ
          MPYU          SNU
          NOOP
          MDAE          PR1
          ARMD'H        PX
          MDAR          PZ
          MPYU          CNU
          NOOP
          MDAE          PR4
          ARMD'H        PZ
          JUMP'I        DRYNG

DTHETA:   NOOP
STHETA:   NOOP
CTHETA:   NOOP
SFI:      NOOP
CFI:      NOOP
PR1:      NOOP
PR4:      NOOP
DMEGA:    NOOP
CEPS:     NOOP
SEPS:     NOOP
CKAPPA:   NOOP
SKAPPA:   NOOP
DLBDA:    NOOP
CMU:      NOOP
SMU:      NOOP
CNU:      NOOP
SNU:      NOOP
MPYU=15400!H
TERMINATE
```

```
EXPUNGE
TITLE CVTXY
ENTRY CVTXY
COMMON CX,CY,CZ,PX,PY,PZ,XX,YY,DSTNC,ALPHA,BETA,GAMMA,SBETA
COMMON F1,F2,THETA
CVTXY:  NOOP
        MDAR            F1
N1:     JPLS            N1+2
        JUMP            BEGIN
        MDAR'L
        Ø
        ARMD            F1
        MDAR            ALPHA       [ALPHA STORED IN BITS Ø-14
        JPSR            $COSA
        ARMD            CALPH       [ COS ALPHA IN BITS Ø TO 14
        MDAR            BETA
        JPSR            $COSA
        ARMD            CBETA       [ COS BETA IN BITS Ø TO 14
        MDAR            GAMMA
        JPSR            $COSA
        ARMD            CGAMA       [ COS GAMMA IN BITS Ø TO 14
        MDAR            BETA
        JPSR            $SINA
        ARMD            SBETA       [ SIN BETA IN BITS Ø TO 14
        MDAR            GAMMA
        JPSR            $SINA
        ARMD            SGAMA       [ SIN GAMMA IN BITS Ø TO 14
        MDAR            DSTNC       [ IN BITS 15 TO 29
        MPYU            CALPH
        NOOP
        ARRS            17
        MDAS            CX
        ARMD            QX          [LAST 6 STEPS COMPUTE QX IN 15-29
        MDAR            DSTNC
        MPYU            CBETA
        NOOP
        ARRS            17
        MDAS            CY
        ARMD            QY          [QY IN BITS 15 TO 29
        MDAR            DSTNC
        MPYU            CGAMA
        NOOP
        ARRS            17
        MDAS            CZ
        ARMD            QZ          [ IN 15 TO 29
```

```
BEGIN:  NOOP
        MDAR            CX
        MPYU            CALPH
        NOOP
        ARRS            20
        NOOP
        ARMD            XMCX
        MDAR            PX
        MPYU            CALPH
        NOOP
        ARRS            20
        NOOP
        MDAS'N          XMCX
        ARMD            XMCX        [15 TO 29
        MDAR            CY
        MPYU            CBETA
        NOOP
        ARRS            20
        NOOP
        ARMD            YMCY
        MDAR            PY
        MPYU            CBETA
        NOOP
        ARRS            20
        NOOP
        MDAS'N          YMCY
        ARMD            YMCY
        MDAR            CZ
        MPYU            CGAMA
        NOOP
        ARRS            20
        NOOP
        ARMD            ZMCZ
        MDAR            PZ
        MPYU            CGAMA
        NOOP
        ARRS            20
        NOOP
        MDAS'N          ZMCZ
        ARMD            ZMCZ
        MDAS            YMCY
        MDAS            XMCX
        ARMD'H          NENNR
        MDAR'H          DSTNC
        ARRS            1
        DIVU            NENNR
        NOOP
        ARMD'H          KKK
```

```
ONE:    MDAR        CX
        MPYU        KKK
        NOOP
        ARMD        CXKKK       [0 TO 28
        MDAR        PX
        MPYU        KKK
        NOOP
        MDAE'N      CXKKK
        MDAE'H      CX
        ARMD        XI
        MDAR        CY
        MPYU        KKK
        NOOP
        ARMD        CYKKK
        MDAR        PY
        MPYU        KKK
        NOOP
        MDAE'N      CYKKK
        MDAE'H      CY
        ARMD        ETA
        MDAR        CZ
        MPYU        KKK
        NOOP
        ARMD        CZKKK
        MDAR        PZ
        MPYU        KKK
        NOOP
        MDAE'N      CZKKK
        MDAE'H      CZ
        ARMD        ZETA
        MDAR        QY
        MPYU        CALPH
        NOOP
        ARMD        XXZL2
        MDAR        ETA
        ARRS        17
        MPYU        CALPH
        NOOP
        MDAE'N      XXZL2
        ARMD        XXZL2
        MDAR        QX
        MPYU        CBETA
        NOOP
        ARMD        XXZL1
        MDAR        XI
        ARRS        17
        MPYU        CBETA
        NOOP
        MDAE'N      XXZL1
        ARMD        XXZL1
        MDAE'N      XXZL2
        DIVU        SGAMA
        NOOP
        ARMD        XX
        MDAR        ZETA
        MDAE'H'N    QZ
```

```
            DIVU            SGAMA
            NOOP
            ARMD            YY
            JMP'I           CVTXY
CALPH:      NOOP
CBETA:      NOOP
CGAMA:      NOOP
SGAMA:NOOP
QX:         NOOP
QY:         NOOP
QZ:         NOOP
XMCX:       NOOP
YMCY:       NOOP
ZMCZ:       NOOP
KKK:        NOOP
NENNR:      NOOP
XI:         NOOP
ETA:        NOOP
ZETA:       NOOP
XXZL1:      NOOP
XXZL2:      NOOP
CXKKK:      NOOP
CYKKK:      NOOP
CZKKK:      NOOP
STORE:      NOOP
ARLS=17500!H
MPYI=15440!H
ARRS=17440!H
MPYU=15400!H
MPYL=15420!H
DIVU=16400!H
TERMINATE
```

```
EXPUNGE
TITLE VSRS
ENTRY VSRS
VSRS:   NOOP
        MDAR'L
        60000
        ARMD            PARX1
        MDAR'L
        70000
        ARMD            PARY1
        MDAR'L
        20000
        ARMD            PARX2
        ARMD            PARY2
        MDAR            DELY
        MDBR            PTR1
        JPSR            $SCHB
RTURN:  JPLS            SAMPL
        MDAR            RXCST
        ARMD            PARY1
        MDAR            RYSML
        ARMD            PARX1
        MDAR            RXMOD
        ARMD            PARY2
        MDAR            RYBIG
        ARMD            PARX2
        MDAR            DELX
        MDBR            PTR1
        JPSR            $SCHB
RURN:   JPLS            SAMPL
        MDAR            RYSML
        ARMD            VXX1
        MDAR            RXCST
        ARMD            VYY1
        MDAR            RYBIG
        ARMD            VXX2
        MDAR            RXMOD
        ARMD            VYY2
BILD:   MDAR'H          VXX1
        MDAR'A'L
        77777!H
        ARMD'H          VXX1
        MDAR            VYY1
        MDAR'A'L
        0!H77777
        MDAE'H          VXX1
        ARMD            POSXY
        BITCR
        0
        MD10'0'L
        10400!H
        MD11'L
        30000!H
        MD07'L
        0
        MD06'L
```

```
            20000
            MD05            POSXY
            MDAR'H          VXX2
            MDAR'A'L
            77777!H
            ARMD'H          VXX2
            MDAR            VYY2
            MDAR'A'L
            0!H77777
            MDAE'H          VXX2
            ARMD            POSXY
            BITCR
            0
            MD10'O'L
            14400!H
            MD05            POSXY
SAMPL:      MDIC'O'L
            40!H
            S5AR'H'F
            MDAR'A'L
            40000
            JPLS            DYPOS
M1:         S5AR'H'F
            MDAR'A'L
            20000
            JPLS            DYNEG
M2:         S5AR'H'F
            MDAR'A'L
            10000
            JPLS            DXPOS
M3:         S5AR'H'F
            MDAR'A'L
            4000
            JPLS            DXNEG
M4:         JUMP            VSRS
DYPOS:      MDAR            DELY
            MDAS'F          1
            ARMD            DELY
            JUMP            M1
DYNEG:      MDAR            DELY
            MDAS'N'F        1
            ARMD            DELY
            JUMP            M2
DXPOS:      MDAR            DELX
            MDAS'F          1
            ARMD            DELX
            JUMP            M3
DXNEG:      MDAR            DELX
            MDAS'N'F        1
            ARMD            DELX
            JUMP            M4
PARX1:      0
PARY1:      0
PARX2:      0
PARY2:      0
RXCST:      0
```

```
RYSML:   Ø
RXMOD:   Ø
RYBIG:   Ø
VYY1:    Ø
VXX1:    Ø
VYY2:    Ø
VXX2:    Ø
DELY:    Ø
DELX:    Ø
PTR1:    NOOP!HPARX1
PTR2:    NOOP!HRYSML
POSXY:   Ø
BITCR=50010!H
MDØ5=25000!H
MDØ6=26000!H
MDØ7=27000!H
MD10=30000!H
MD11=31000!H
TERMINATE
```

```
EXPUNGE
TITLE SCHB
ENTRY    SCHB
SCHB:    NOOP
         ARMD              DELY
         ARMD'N            DDY
         BRMD              SAVE
         MDAR'I            SAVE
         ARMD              XX1
         MDAR'X'I          SAVE
         ARMD              YY1
         MDAR'X'I          SAVE
         ARMD              XX2
         MDAR'X'I          SAVE
         ARMD              YY2
         MDAR              DELY
         JPLS              DECEL
         JUMP              QUICK
DECEL:   ARAR'H'F
         JPAN              .+2
         JUMP              PSTIV
         MDAR'H            YY2
         JPAN              F1
         MDAR'H            YY1
         JPAN              F111
         MDAE'H'N          YY2
         JPAN              F111
FØ1:     MDAR              YY1
         ARMD              YYBIG
         MDAR              YY2
         ARMD              YYSML
         MDAR              XX1
         ARMD              XXMOD
         MDAR              XX2
         ARMD              XXCST
         JUMP              STF2
F1:      MDAR'H            YY1
         JPAN              F11
         JUMP              FØ1
F11:     MDAR'H'N          YY2
         MDAE'H            YY1
         JPAN              F111
         JUMP              FØ1
F111:    MDAR              YY2
         ARMD              YYBIG
         MDAR              YY1
         ARMD              YYSML
         MDAR              XX2
         ARMD              XXMOD
         MDAR              XX1
         ARMD              XXCST
STF2:    MDAR'N            YYSML
         ARBR'F
         MDAR'N            YYBIG
         ARMD              YYSML
         BRMD              YYBIG
```

```
        MDAR'N          XXCST
        ARBR'F
        MDAR'N          XXMOD
        ARMD            XXCST
        BRMD            XXMOD
        MDAR'H          YYSML
        JPAN            G1
        MDAR'L
        37777!H
        MDAE'H'N        YYSML
        MDAE'H'N        DDY
        JPAN            STP51
        MDAE'H'N        DDY
        JPAN            STP51
        JUMP            G2
G1:     ARAR'N'F
        MDAE'H'N        DDY
        JPAN            G11
        JUMP            G2
G11:    MDAE'L
        37777!H
        MDAE'H'N        DDY
        JPAN            STP51
G2:     MDAR'H          YYBIG
        JPAN            G3
        MDAR'L
        37777!H
        MDAE'H'N        YYBIG
        MDAE'H'N        DDY
        JPAN            G21
        MDAE'H'N        DDY
        JPAN            G22
        JUMP            STF4
G22:    ARAR'N'F
        DIVL            DDY
        NOOP
        ARMD            FR1
        MDAR'L
        JUMP            STF36
        ARMD            KIMP
        JUMP            STF3
G21:    ARAR'N'F
        DIVL            DDY
        NOOP
        ARMD            FR1
        MDAR'L
        NOOP
        ARMD            KIMP
        JUMP            STF3
G3:     ARAR'N'F
        MDAE'H'N        DDY
        JPAN            G31
        JUMP            STF4
G31:    MDAE'L
        37777!H
        MDAE'H'N        DDY
```

```
            JPAN        G32
            JUMP        STF4
G32:        ARAR'N'F
            DIVL        DDY
            NOOP
            ARMD        FR1
            MDAR'L
            JUMP        STF36
            ARMD        KIMP
STF3:       MDAR'H'N    YY1
            ARRS        1
            ARMD        DABVE
            MDAR'H'N    YY2
            ARRS        1
            MDAE'N      DABVE
            ARMD        DZAE
            ANAR'N      DZAE
            ARMD        DNOM
            MDAR'H'N    XX1
            ARRS        1
            ARMD        DBLOW
            MDAR'H'N    XX2
            ARRS        1
            MDAE'N      DBLOW
            ARMD        DNR
KIMP:       NOOP
            MDAR        DDY
            MPYU        DNR
            NOOP
            ARMD        NUMR
            ANAR'N      NUMR
            MDAE'N      DNOM
            JPAN        HØ
            MDAR        NUMR
            ARRS        1
            DIVU        DZAE
            NOOP
            ARMD'H      XCH1
            MPYL        FR1
            NOOP
            ARAE
            ARMD        XCH2
            MDAR'H      XXMOD
            MDAE'N      XCH1
            MDAE'N      XCH1
            MDAE'N      XCH2
            ARMD'H      XXMOD
            JUMP        STF37
HØ:         MDAR'N      NUMR
            DIVU        DZAE
            NOOP
            ARMD'H      XCH1
            MPYL        FR1
            NOOP
            MDAE'H      XXMOD
            MDAE        XCH1
            ARMD'H      XXMOD
            JUMP        STF37
```

```
STF36:  MDAR            FR1
        MPYL            DDY
        NOOP
        ARAR'H'F
        MPYU            DNR
        NOOP
        ARMD            NUMR
        ANAR'N          NUMR
        MDAE'N          DNOM
        JPAN            H2
        MDAR            NUMR
        ARRS            1
        DIVU            DZAE
        NOOP
        ARMD            XCH1
        MDAR            XXMOD
        MDAS'N          XCH1
        MDAS'N          XCH1
        ARMD            XXMOD
        JUMP            STF37
H2:     MDAR'N          NUMR
        DIVU            DZAE
        NOOP
        MDAS            XXMOD
        ARMD            XXMOD
STF37:  MDAR'F          37777
        ARMD            YYBIG
        MDAR            DDY
        ARAS'F
        MDAS            YYSML
        ARMD            YYSML
        JUMP            TORT
STF4:   NOOP
        MDAR            DDY
        ARAS'F
        MDAS            YYBIG
        ARMD            YYBIG
        MDAR            DDY
        ARAS'F
        MDAS            YYSML
        ARMD            YYSML
TORT:   MDAR'N          YYSML
        ARBR'F
        MDAR'N          YYBIG
        ARMD            YYSML
        BRMD            YYBIG
        MDAR'N          XXCST
        ARBR'F
        MDAR'N          XXMOD
        ARMD            XXCST
        BRMD            XXMOD
        JUMP            DORT
PSTIV:  MDAR'H          YY2
        JPAN            A1
        MDAR'H          YY1
        JPAN            A111
```

```
        MDAE'H'N        YY2
        JPAN            A111
A01:    MDAR            YY1
        ARMD            YYBIG
        MDAR            YY2
        ARMD            YYSML
        MDAR            XX1
        ARMD            XXMOD
        MDAR            XX2
        ARMD            XXCST
        JUMP            STP2
A1:     MDAR'H          YY1
        JPAN            A11
        JUMP            A01
A11:    MDAR'H'N        YY2
        MDAE'H          YY1
        JPAN            A111
        JUMP            A01
A111:   MDAR            YY2
        ARMD            YYBIG
        MDAR            YY1
        ARMD            YYSML
        MDAR            XX2
        ARMD            XXMOD
        MDAR            XX1
        ARMD            XXCST
STP2:   MDAR'H          YYSML
        JPAN            B1
        MDAR'L
        37777!H
        MDAE'H'N        YYSML
        MDAE'H'N        DELY
        JPAN            STP51
        MDAE'H'N        DELY
        JPAN            STP51
        JUMP            B2
B1:     ARAR'N'F
        MDAE'H'N        DELY
        JPAN            B11
        JUMP            B2
B11:    MDAE'L
        37777!H
        MDAE'H'N        DELY
        JPAN            STP51
B2:     MDAR'H          YYBIG
        JPAN            B3
        MDAR'L
        37777!H
        MDAE'H'N        YYBIG
        MDAE'H'N        DELY
        JPAN            B21
        MDAE'H'N        DELY
        JPAN            B22
        JUMP            STP4
B22:    ARAR'N'F
        DIVL            DELY
```

```
        NOOP
        ARMD            FR1
        MDAR'L
        JUMP            STP36
        ARMD            CIMP
        JUMP            STP3
B21:    ARAR'N'F
        DIVL            DELY
        NOOP
        ARMD            FR1
        MDAR'L
        NOOP
        ARMD            CIMP
        JUMP            STP3
B3:     ARAR'N'F
        MDAE'H'N        DELY
        JPAN            B31
        JUMP            STP4
B31:    MDAE'L
        37777!H
        MDAE'H'N        DELY
        JPAN            B32
        JUMP            STP4
B32:    ARAR'N'F
        DIVL            DELY
        NOOP
        ARMD            FR1
        MDAR'L
        JUMP            STP36
        ARMD            CIMP
STP3:   MDAR'H          YY1
        ARRS            1
        ARMD            DABVE
        MDAR'H          YY2
        ARRS            1
        MDAE'N          DABVE
        ARMD            DZAE
        ANAR'N          DZAE
        ARMD            DNOM
        MDAR'H          XX1
        ARRS            1
        ARMD            DBLOW
        MDAR'H          XX2
        ARRS            1
        MDAE'N          DBLOW
        ARMD            DNR
CIMP:   NOOP
        MDAR            DELY
        MPYU            DNR
        NOOP
        ARMD            NUMR
        ANAR'N          NUMR
        MDAE'N          DNOM
        JPAN            CØ
        MDAR            NUMR
        ARRS            1
```

```
        DIVU            DZAE
        NOOP
        ARMD'H          XCH1
        MPYL            FR1
        NOOP
        ARAE
        ARMD            XCH2
        MDAR'H          XXMOD
        MDAE'N          XCH1
        MDAE'N          XCH1
        MDAE'N          XCH2
        ARMD'H          XXMOD
        JUMP            STP37
C0:     MDAR'N          NUMR
        DIVU            DZAE
        NOOP
        ARMD'H          XCH1
        MPYL            FR1
        NOOP
        MDAE'H          XXMOD
        MDAE            XCH1
        ARMD'H          XXMOD
        JUMP            STP37
STP36:  MDAR            FR1
        MPYL            DELY
        NOOP
        ARAR'H'F
        MPYU            DNR
        NOOP
        ARMD            NUMR
        ANAR'N          NUMR
        MDAE'N          DNOM
        JPAN            C2
        MDAR            NUMR
        ARRS            1
        DIVJ            DZAE
        NOOP
        ARMD            XCH1
        MDAR            XXMOD
        MDAS'N          XCH1
        MDAS'N          XCH1
        ARMD            XXMOD
        JUMP            STP37
C2:     MDAR'N          NUMR
        DIVU            DZAE
        NOOP
        MDAS            XXMOD
        ARMD            XXMOD
STP37:  MDAR'F          37777
        ARMD            YYBIG
        MDAR            DELY
        ARAS'F
        MDAS            YYSML
        ARMD            YYSML
        JUMP            DORT
```

```
STP4:   NOOP
        MDAR            DELY
        ARAS'F
        MDAS            YYBIG
        ARMD            YYBIG
        MDAR            DELY
        ARAS'F
        MDAS            YYSML
        ARMD            YYSML
DORT:   MDAR            XXCST
        ARMD'X'I        SAVE
        MDAR            YYSML
        ARMD'X'I        SAVE
        MDAR            XXMOD
        ARMD'X'I        SAVE
        MDAR            YYBIG
        ARMD'X'I        SAVE
        MDAR'F          0
        JUMP'I          SCHB
STP51:  MDAR'F          1
        JUMP'I          SCHB
QUICK:  MDAR            XX1
        ARMD'X'I        SAVE
        MDAR            YY1
        ARMD'X'I        SAVE
        MDAR            XX2
        ARMD'X'I        SAVE
        MDAR            YY2
        ARMD'X'I        SAVE
        MDAR'F
        JUMP'I          SCHB
XX1:    0
XX2:    0
XXMOD:  0
XXCST:  0
YY1:    0
YY2:    0
YYBIG:  0
YYSML:  0
DELY:   0
FR1:    0
DABVE:  0
DZAE:   0
DNOM:   0
DBLOW:  0
DNR:    0
NUMR:   0
NR1:    0
DAT1:   0
DAT2:   0
DAT3:   0
DAT4:   0
```

```
XCH1:    0
XCH2:    0
DDY:     0
SAVE:    0
MPYU=15400!H
MPYL=15420!H
DIVU=16400!H
DIVL=16420!H
ARRS=17440!H
TERMINATE
```

```
EXPUNGE
TITLE ATODE
ENTRY ATODE
ATODE:  NOOP
        MDAR'L
        20000!H00020
        JPSR            $HEBEL
        MDAR'A'L
        00000!H77777
        ARMD            XVAL
        MDAR'L
        20000!H00040
        JPSR            $HEBEL
        MDAR'A'L
        00000!H77777
        ARMD            YVAL
        JUMP'I          ATODE
XVAL:   0
YVAL:   0
TERMINATE
```

```
EXPUNGE
TITLE HEBEL
ENTRY HEBEL
HEBEL:  NOOP
        ARMD            G0+1
        MDAR'L
        NOOP!HTAFEL
        ARMD            TAFEL
        MD07'H          MFSCL
        MD11'L
        77777!H00000
G0:     MDAR'L
        0
        AR11'F
        MDAR'L
        0!H77766
        ARMD            CTR
        MDBR            MFSCL
G1:     BRAR'F
        SKLA            40
        JUMP            DOWN
G2:     MDAS'X'I        TAFEL
        ARBR'F
        AR07'F'H
        MDAR'X          CTR
        JPLS            G1
        JUMP            OUT
DOWN:   MDAS'N'I        TAFEL
        MDAS'X'I        TAFEL
        ARBR'F
        AR07'F'H
        MDAR'X          CTR
        JPLS            G1
        JUMP            OUT
OUT:    BRAR'H'N'F
        DIVI            13400
        NOOP
        JUMP'I          HEBEL
TAFEL:  0!HTAFEL
        37777
        20000
        10000
        4000
        2000
        1000
        400
        200
        100
        40
MFSCL:  40000
DIVI=16440!H
MD11=31000!H
AR07=27100!H
AR11=31100!H
MD10=30000!H
MD07=27000!H
CTR:    NOOP
TERMINATE
```

BIBLIOGRAPHY

1. Adage graphics terminal
   Amos resident monitor
   Programmer's reference manual

2. Adage graphics terminal
   Digitial processor
   Subsystem programming specifications

3. Adage graphics terminal
   Function switches
   Programming specifications

4. Adage graphics terminal
   Graphics coordinate transformation array
   Subsystem programming specifications

5. Adage graphics terminal
   Programming instruction manual

6. Adage graphics terminal
   Vector generator
   Subsystem programming specifications

7. Albert, A. A., Solid Analytic Geometry, 1st ed., p. 103-110, McGraw-Hill, 1949.

8. Cotton, I. W. and F. S. Greatorex, "Data structures and techniques for remote computer graphics", AFIPS Conference Proceedings, p. 533, 1968.

9. Naval Missile Center, Janair Report No. 680713, Janair Vertical Contact Analog Display Evaluation Program, by K. D. Cross and A. C. Bittner, 31 January 1969.

10. Dersch, W. C. and R. T. Johnson, "Computer-managed display system for advanced commerical transports", Recent Advances in Display Media.

11. Desens, R. B., "Computer processing for display of three dimensional structures", Eng. Thesis, Naval Postgraduate School, Monterey, California, 1969.

12. Hagan, T. G., R. J. Nixon and L. J. Schaefer, "The adage graphics terminal", AFIPS Conference Proceedings, p. 747, 1968.

13. Kubert, B., J. Szabo and S. Giulieri, "The perspective representations of functions of two variables", Journal of the ACM, Vol. 15, No. 2.

14. Sutherland, I. E., "Sketchpad: A man machine graphical communication system", Lincoln Lab/MIT Technical Report No. 296, 1963.

15. Vlahos, P., "The three-dimensional display. Its cues and techniques", Information Display, Nov/Dec 1965.

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Documentation Center Cameron Station Alexandria, Virginia 22314	20
2.	Library Naval Postgraduate School Monterey, California 93940	2
3.	Associate Professor Mitchell L. Cotton Department of Electrical Engineering Naval Postgraduate School Monterey, California 93940	1
4.	KptLt Willi K. Krauss 3 Lohrbergstr. Bonn, Germany	1
5.	Commander Naval Ordnance Systems Command Navy Department Washington, D. C. 20360	1

Security Classification

DOCUMENT CONTROL DATA - R & D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1. ORIGINATING ACTIVITY (Corporate author)	2a. REPORT SECURITY CLASSIFICATION
Naval Postgraduate School Monterey, California 93940	Unclassified
	2b. GROUP

3. REPORT TITLE

Application of Display in Flight Vehicle Mission Performance Evaluation

4. DESCRIPTIVE NOTES (Type of report and inclusive dates)
Master's Thesis; October 1969

5. AUTHOR(S) (First name, middle initial, last name)

Willi Konrad Alois Krauss

6. REPORT DATE	7a. TOTAL NO. OF PAGES	7b. NO. OF REFS
October 1969	82	15

8a. CONTRACT OR GRANT NO.	9a. ORIGINATOR'S REPORT NUMBER(S)
b. PROJECT NO.	
c.	9b. OTHER REPORT NO(S) (Any other numbers that may be assigned this report)
d.	

10. DISTRIBUTION STATEMENT

This document has been approved for public release and sale; its distribution is unlimited.

11. SUPPLEMENTARY NOTES	12. SPONSORING MILITARY ACTIVITY
	Naval Postgraduate School Monterey, California 93940

13. ABSTRACT

With the increase in speed of moving vehicles there is a growing need for rapid, easy to interprete display of information concerning its dynamic state.

With growing equipment stability a remedy can grow out of research and application of computer aided information presentation, especially the production of proper displays. While all information can be made to correspond to familiar schemes as close as desired new approaches to the general problem must be sought to find an improved information presentation. This study is concerned with the application of computer generated graphics for one or more display consoles for aid in the operation of the vehicle. In the first chapter a major aspect of the general program is chosen for a demonstration: a 'flight vehicle to destination' situation is subjected to an experimental investigation.

A general purpose computer with a graphics terminal that has become available just recently has been chosen for the study. Thereby implicitly a measure of the capability to perform a real time on line job is obtained.

DD FORM 1 NOV 65 1473 (PAGE 1)
S/N 0101-807-6811

Security Classification

A-31408

14. KEY WORDS	LINK A		LINK B		LINK C	
	ROLE	WT	ROLE	WT	ROLE	WT
Management program						
Graph complexity						
Projection and rotation						
Pseudo 3D superimposed structure						
Vector generation/AGT10						